IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received 11 April 2025, accepted 27 May 2025, date of publication 2 June 2025, date of current version 9 June 2025.

*Digital Object Identifier 10.1109/ACCESS.2025.3575494*

SURVEY

# A Survey on Data Plane Security in Software-Defined Networks: Toward Adaptive Security of Data Planes

**AMINA TANKOVIC[1], (Member, IEEE), EMIR DERVISEVIC[1], MIRALEM MEHIC[1,2], (Member, IEEE), AND ENIO KALJIC[1]**

[1]Department of Telecommunications, Faculty of Electrical Engineering, University of Sarajevo, 71000 Sarajevo, Bosnia and Herzegovina
[2]Department of Telecommunications, VSB—Technical University of Ostrava, 708 00 Ostrava, Czechia

Corresponding author: Amina Tankovic (atankovic1@etf.unsa.ba)

This work was supported in part by the Ministry of Science, Higher Education and Youth of Canton Sarajevo, Bosnia and Herzegovina under Grant 27-02-35-35137-30/22 and Grant 27-02-35-33081-9/24.

**ABSTRACT** Software-Defined Networking (SDN) is the actual approach in the network design, based on separating the control and data plane. Such architectural model has brought improvements in terms of network monitoring, management and troubleshooting, but has also increased risks related to network security. Security attacks can occur at all SDN layers and disrupt part or the entire network. Existing research is mostly focused on the security of the control plane, since it contains all control logic of SDN networks and thus represents their main part. Although the data plane has many vulnerabilities and can also be a significant source of security threats towards the control plane, it is only partially covered in existing research, without enough details related to differences between methods and implementation techniques which provide security enhancement. In this paper, we present a comprehensive survey on security of the data plane, focusing on the latest advanced solutions. The survey starts with an overview of attacks, threats and affected security attributes in the data plane, classified using common security models: STRIDE, CIA and AAA. After that, we present a detailed analysis of solutions explored in the literature, including the methods used for security enhancement, implementation techniques, experimental environments, their contributions in terms of vulnerabilities that they address, performance analysis and limitations. Through this analysis, we introduce the concept of adaptive security and select several mechanisms which can be used to achieve it. Additionally, we propose possible combinations of presented mechanisms to provide strong, comprehensive solution which should adapt to dynamics of network, attackers and users, and in that way protect the network from different threats and also satisfy the requirements of services which need different levels of security.

**INDEX TERMS** Adaptivity, data plane, security, software-defined networks.

## I. INTRODUCTION

The development of new services has brought challenges in the design and administration of networks. IoT/IoE (Internet of Things/Internet of Everything) [1], cloud services, big data [2] and other popular applications in 5G/6G networks require high availability, throughput and dynamic management of a large number of devices. Traditional networks cannot completely meet these requirements because they are very difficult for modification. Their modifications require time and specific knowledge about configuring devices from different vendors [3]. Even minimal changes, such as protocol change (e.g. using IPv6 (Internet protocol version 6) instead of IPv4 (Internet protocol version 4)), in traditional networks can cause a problem [4]. In order to make the network management easier, a new paradigm called software-defined networking (SDN) has been developed.

The concept on which software-defined networks are based was introduced in the mid-1990s, when the need for increased network programmability arose [5]. Today, the term ''SDN''

The associate editor coordinating the review of this manuscript and approving it for publication was Salekul Islam.

primarily refers to a network architecture where the control and data plane are separated from each other. By breaking vertical integration of these planes, devices in the data plane started to perform only simple forwarding function. All control logic that defines forwarding rules for each switch is implemented in a logically centralized controller, programmed by appropriate applications in the application plane [4], [6]. An example of simplified SDN architecture is shown in Fig. 1.

Separation of the control and data plane has brought numerous advantages for next-generation networks, in terms of flexibility, performance and management, including interoperability between solutions developed for different network generations. However, this kind of architecture has also brought risks in terms of security, which is not yet a built-in feature [7]. Since SDN is already in use in 5G networks, and is expected to be used in 6G and next-generation networks (e.g. QKD (Quantum Key Distribution) networks, for which there are already proposed integrations with SDN), it is necessary to be aware of possible security threats that can occur in such architecture, in order to enhance security of future networks [8], [9].

Various attacks are possible at all SDN layers (i.e. planes) represented in Fig. 1. Since the controller contains the logic through which it manages all network devices, it represents "the brain" of the entire network, whose operation depends on the security and correct operation of the controller. For this reason, the research so far has mostly focused on the security of the control plane. Although the data plane has not been explored to the same extent, it also has many vulnerabilities and should be carefully analysed due to the following reasons:

- Switches in the data plane are front-line network elements, which makes them directly exposed to potential attackers.
- The data plane represents network fabric which connects all users, including malicious ones, which can perform various types of attacks and compromise legitimate users.
- The data plane transmits control information between switches and the control plane, thus it can be a collateral target of attacks on the controller.

This can be additionally explained using the example shown in Fig. 1. It is assumed that in-band signalling is used in the network (i.e. that the control and data traffic are sent over the same physical link), since it is more often than out-of-band signalling in practical implementations due to lower cost [10]. In this case, malicious traffic can commingle with traffic that is being exchanged between legitimate users, switches, or towards/from the controller. Additionally, security of the data plane will be especially important in 6G networks, since they are expected to bring the concept of self-organizing networks [11] which will make them less centralized and focused more on the data plane, which should be fast, adaptive and secure.

The attacks that can occur at the data plane are mostly not specific to SDN, since it contains forwarding devices which are also present in traditional networks [12]. In terms of the southbound interface, threats are possible mostly due to the vulnerabilities of the OpenFlow protocol, which defines communication between the controller and the data plane [13]. Vulnerabilities of both the data plane and the southbound interface allow the attackers to perform various types of MitM (Man-in-the-Middle) and DoS (Denial-of-Service) attacks. Some of them (e.g. flooding) can wreak havoc in the network, while some others (such as eavesdropping or spoofing) are only the first step towards more dangerous attacks. Therefore, it is necessary to explore all these security risks and possible solutions for them, with the aim of exploring state-of-the-art and determining new approaches for further security enhancements.

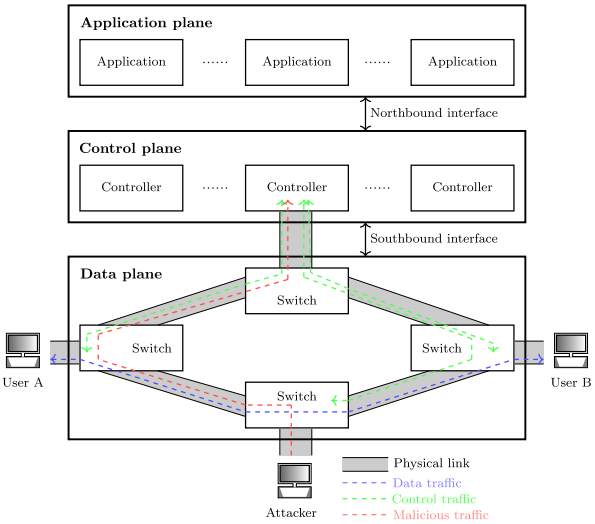


**FIGURE 1. An example of a vulnerable data plane within simplified software-defined network architecture. Data traffic can be exchanged between users or switches and control traffic is exchanged between switches and the controller. Attackers can access any switch in the data plane and send malicious traffic towards some switch or the controller, and in that way compromise all devices and legitimate traffic on that path.**

### A. BACKGROUND

This section contains definitions and explanations of terms that are important for the analysis presented in the rest of this paper. In order to give comprehensive and detailed overview of the data plane security, several terms should be introduced and clearly distinguished:

- vulnerability - a flaw or weakness in the network that could be exploited by malicious party to violate network security;
- security threat - a potential violation of security of the data or the network, including information leakage, denial of service, different modifications and spoofings;
- attack - realisation of a threat, in a passive or active way;
- security attribute - main properties of the data and the network that need to be secured in order to provide secure communication.

It can be stated that vulnerabilities are sources of potential violations of security attributes, which can be exploited by

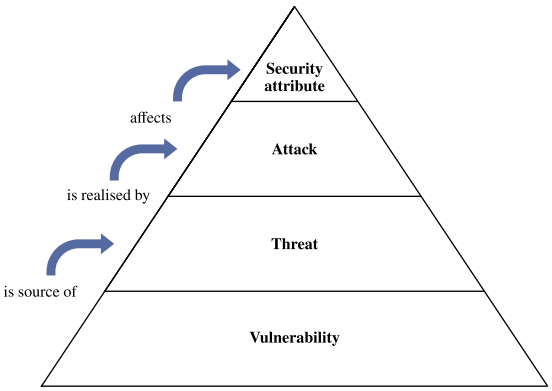


**FIGURE 2.** Hierarchy of security problems: Vulnerabilities represent sources of potential violations of security, i.e. threats. Threats become realised by attacks. Attacks are performed with the aim to affect crucial security attributes.

malicious parties to perform attacks and in that way realise threats (Fig. 2). Therefore, security solutions need to address vulnerabilities, and in that way prevent, mitigate or at least detect attacks and potential threats related to them.

When it comes to solutions analysis, several important terms need to be clarified:

- security method - different ways of the data plane security enhancement (e.g. encryption, monitoring, classifications etc.);
- implementation technique - different ways of implementation of proposed methods (e.g. middleboxes, applications/modules, programmable data plane or new framework/scheme/architectural model);
- experimental environment - hardware or software environment used for implementation and testing (e.g. simulators, different types of switches and controllers, reconfigurable hardware);
- contribution - measure of enhancement that the proposed solution brings (e.g. detection, prevention, mitigation);
- performance analysis - impact of the used security method or implementation technique on network performance (e.g. latency, bandwidth);
- scalability - ability of a security solution to keep achieved contribution and performance level in larger networks;
- limitations - constraints of a security solution related to security and performance level that can be achieved due to the used security method or implementation technique (including experimental environment).

Another important concept that needs to be considered in security solutions is adaptivity, which can be defined as the ability of a system to adapt to changes related to security, including the following sources of changes (Fig. 3):

- dynamics of an user - changes in the level of security that different users require, usually related to a specific service (e.g. sending sensitive information which requires the highest level of confidentiality and integrity protection);
- dynamics of an attacker - changes in the type of attack that malicious user performs (e.g. series of different attacks which are often performed successively);
- dynamics of the network - changes that arose as a consequence of network failures (e.g. link or switch failure) or network operation, including basic functions or advanced functions (e.g. some security function which can lead to temporary lack of required security resources, such as specific encryption keys).

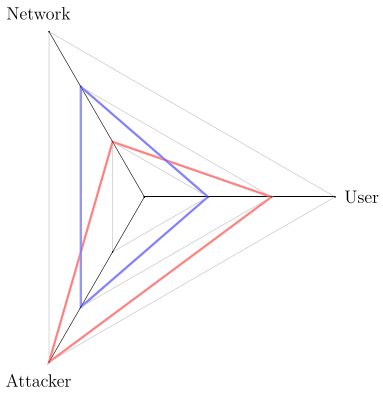


**FIGURE 3.** Three sources of dynamics/changes which need to be considered in security solutions represented using spider chart: A particular security solution (a colored triangle) should respond to each dynamics to a certain extent.

Considering modern applications' purposes and requirements, and a variety of existing attacks, it can be stated that aforementioned dynamics are crucial factors in terms of security. Therefore, it will be important to find particular methods and implementation techniques through which these dynamics can be addressed, achieving adaptive security of the data plane.

### B. RELATED WORK

Although the data plane can be a critical part of SDN networks when it comes to security, it hasn't attracted much attention in the existing literature. Related surveys mostly cover the security of the entire SDN network, focusing more on the security of the control plane. Security of the data plane is only partially covered, mainly without clear distinction between possible attacks, threats and vulnerabilities, and without deeper analysis of different methods and implementation techniques for proposed security solutions. Also, only few of them discuss the possibility of adaptivity in terms of security, considering AI (Artificial Intelligence) methods to adapt to more complex security scenarios.

Papers [14], [15], [16], [17], [18], [19] give surveys on security at all layers of SDN networks, using different approaches. Authors in [14] present possible attacks and affected security dimensions according to ITU-T recommendations. Also, they analyse a number of security solutions and present some differences between them in terms of security type, implementation framework and cost. However, their survey is mainly focused on other SDN layers, thus it doesn't contain enough important details related to the

**TABLE 1. Overview of related surveys considering their analyses of security problems (attacks, threats, vulnerabilities, affected security attributes and usage of any of common security models), solutions (methods, implementation technique, experimental environment, contribution, performance analysis, scalability, limitations and possibility of adaptive security), focus on the data plane and possible future directions: ✓ indicates the topic is well covered, ∗ indicates the topic is partially covered.**

| Reference | Security problem | | | | | Security solution | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | Attacks | Threats | Vulnerabilities | Security attributes | STRIDE/AAA/CIA | Methods | Implementation | Experimental environment | Contribution | Performance analysis | Scalability | Limitations | Adaptivity | Focus on the data plane | Future directions |
| Ahmad et al. [14] | ✓ | | | ∗ | ✓ | ∗ | ∗ | | ∗ | | | | ∗ | | ∗ |
| Alsmadi et al. [15] | ∗ | ✓ | | ∗ | ✓ | ∗ | ∗ | | ∗ | | ∗ | ∗ | ∗ | | ∗ |
| Chica et al. [16] | ✓ | ∗ | ✓ | | | ✓ | ∗ | | ✓ | | | | ∗ | | ∗ |
| Scott et al. [17] | ✓ | ∗ | ∗ | | | ∗ | ∗ | | ✓ | ∗ | ∗ | | ∗ | | ∗ |
| Rahouti et al. [18] | ✓ | | | | | ✓ | ∗ | ∗ | ✓ | | | | ∗ | | ∗ |
| Farooq et al. [19] | ✓ | ∗ | ∗ | | | ∗ | | | ✓ | ∗ | | | | | ∗ |
| Deb et al. [20] | ✓ | | ✓ | ✓ | | ✓ | ✓ | ∗ | ✓ | ∗ | | ✓ | ∗ | | ∗ |
| Jimenez et al. [21] | ✓ | ✓ | ∗ | | ✓ | ✓ | ∗ | ∗ | ∗ | ∗ | | | | | ∗ |
| Dayal et al. [22] | ✓ | | | ✓ | | ∗ | ∗ | | ✓ | | | | ∗ | | |
| Abdi et al. [23] | ✓ | ✓ | | ∗ | ✓ | ✓ | ∗ | ✓ | ✓ | | ∗ | ∗ | ∗ | | ∗ |
| Shahzad et al. [24] | ∗ | | | | | ∗ | | | ∗ | | ✓ | ✓ | ∗ | | ∗ |
| Shaghaghi et al. [25] | ✓ | | ✓ | | | ∗ | ∗ | ∗ | ∗ | ✓ | | ✓ | | ∗ | ∗ |
| Dargahi et al. [26] | ✓ | | ✓ | | | ∗ | ∗ | | ∗ | ∗ | ∗ | | | ∗ | ∗ |
| Alhaj et al. [27] | ✓ | ∗ | | | | | | | | | | | | | |
| Pradhan et al. [28] | ✓ | | ∗ | | | ∗ | | | | | | | | | |
| Krishnan et al. [29] | ∗ | ∗ | | ✓ | | ∗ | | | | | | | | | ∗ |
| Iqbal et al. [30] | ✓ | | | ✓ | | ∗ | | | | | | | | | |
| Dabbagh et al. [31] | ∗ | | ✓ | | | ∗ | | | | | | | | | |
| Mahar et al. [32] | ∗ | | | | | ∗ | | | | | | | | | |
| This survey | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |

data plane security. In [15], authors have done analysis in a slightly different way, focusing more on the aim of each attack by considering threats specified by STRIDE (Spoofing, Tampering, Repudiation, Information disclosure, Denial of service, Elevation of Privilege) model. Since they also analyse the entire SDN architecture, they haven't made a clear distinction between attacks on different layers. Besides some potential threats, authors in [16] also present vulnerabilities that cause each identified attack. Also, they mention some issues in existing security solutions related to some specific methods, attacks and environments. Similar analyses related to the data plane are given in [17], [18], and [19]. These surveys only partially cover the security of the data plane, but they give comprehensive analyses of SDN in general, its architecture, security and applications.

Authors in [20], [21], [22], and [23] also analyse security at all SDN layers, but they present more details related to the data plane and possible solutions. In [20] is given a very detailed overview of potential attacks, vulnerabilities, and proposed solutions on different SDN layers, summarized per security features/dimensions. Authors in [21] analyse security problems without clear separating attacks and threats, but have given a more detailed overview of proposed solutions compared to previous surveys. Although their solution analysis is comprehensive, they have focused more on the used methods rather than the implementation techniques. They have emphasized some solutions that use P4 language, but have not mentioned implementation techniques for all presented solutions. Authors in [22] identify a number of attacks at all SDN layers, but focus on solutions only for DoS/DDoS attacks. Observed solutions use various methods for mitigation and detection of these attacks. Some of them are based on machine learning which allows them to adapt to network conditions. Therefore, these solutions provide adaptivity in this way. In [23] authors present a comprehensive review of traditional, AI and MTD

(Moving Target Defense) approaches to security solutions for SDN control and data plane. This survey contains very detailed analysis of potential attacks in SDN, with different methods, implementation techniques and simulation tools used for solutions. Since they are focused on AI and MTD techniques, they also discuss adaptivity, but only in terms of these techniques. Traditional and AI approaches are also analysed in [24]. However, authors in this paper focused more on blockchain as a promising solution for securing SDN architecture.

Papers [25] and [26] are specifically focused on the security of the data plane. Authors in [25] have focused on comparison of two specific solutions, SPHINX and WedgeTail, which satisfies the requirements that they identified as mandatory for security solutions. In [26] is given a survey on the security of stateful data planes. Authors present security issues that are inherited from traditional SDN, and those that are introduced by increased programmability. However, they are focused only on attacks that are specific for stateful data planes, and describe some possible directions for their mitigation, which cannot be applied to the data plane in general.

Some less comprehensive overviews are given in [27], [28], [29], [30], [31], [32]. In these papers, authors mostly define only potential attacks that can occur in data planes, with not enough detailed analysis of security solutions that have already been proposed in the existing literature. Some of them provide basic solutions which can be used, as a possible direction for security improvement, without giving any details in terms of implementation.

In Table 1 is given an overview of related work. Most of the existing surveys cover the data plane only partially, as a part of research on security of the entire SDN architecture. Hence, none of them give complete analysis of security problems and solutions related to the data plane. Very few of them consider adaptivity, mostly only in terms of AI methods. Also, future work in analysed papers usually present only simple directions which address gaps in the existing literature, without proposing new possible solutions for further security enhancement.

### C. CONTRIBUTIONS

The aim of this paper is to address previously mentioned gaps in the existing literature providing a detailed review of security problems and solutions which is focused exclusively on the data plane security. Surveys so far do not provide sufficient analysis of potential security problems in the data plane, since they do not provide relations between sources of these problems, ways of their realisation and consequences that they can bring. In fact, they are usually mainly focused on the specific realisation of security problems (i.e. attacks). However, knowing the sources of security problems (i.e. vulnerabilities) can help in better analysis of potential solutions for them. Research on security solutions presented in surveys so far is not comprehensive enough and does not include all important details. Therefore, there is a need for more detailed review of security problems and solutions in the data plane, which can help to find approaches to address different complex security situations that can occur in the network. This survey not only provides a comprehensive analysis of the existing data plane security research, but also gives possible directions for further security enhancement which are not considered in existing surveys.

The main contributions of this paper are as follows:

- Identification and clear distinction of:
  - attacks that can occur at the data plane and the southbound interface of SDN networks;
  - threats that are realised by each of these attacks, according to the STRIDE model [33];
  - vulnerabilities that exist in the data plane and make each of the identified attacks possible;
  - security attributes that are affected by each of these attacks, according to the CIA (Confidentiality, Integrity, Availability) or AAA (Authentication, Authorization, Accounting) models [34].
- Review and analysis of existing security solutions, including methods, implementation techniques, experimental environment, contributions, performance and scalability analysis and limitations;
- Review and analysis of mechanisms that can be used to achieve adaptive security of the data plane;
- Future research directions for further improvements of the data plane security, considering the proposed adaptivity enablers and security methods that are not sufficiently explored yet in the existing literature, in order to address user, attacker and network dynamics.

### D. PAPER ORGANIZATION

The rest of this paper is organised as follows. Section II presents attacks that can occur at the data plane, the aim and brief description of each of them. It also gives their relations with possible threats according to the STRIDE model, and security attributes that they affect, classified using CIA and AAA models. Section III gives an overview of existing solutions, with analysis of the methods that are used, implementation techniques, experimental environments, contributions, performance analysis, limitations and possible improvements. It also presents relations of these solutions with vulnerabilities that they address. Section IV proposes mechanisms that can be used to achieve adaptive security in different ways and describes how they are used in the existing literature. Section V presents possible directions for future enhancement of the data plane security, considering the proposed adaptivity enablers and different security methods. Section VI gives a conclusion of comprehensive analysis presented in this paper.

## II. ATTACKS AND THREATS IN THE DATA PLANE

Existing literature related to data plane security generally does not make a difference between terms ''attack'', ''threat'' and ''vulnerability''. They are often used interchangeably, or are grouped under the term ''security risk/issue/problem''. Distinction between these terms and their classification will

help in better understanding the causes of security problems in data planes, and consequences that they can lead to. This will also help in better understanding of the proposed security solutions, and also finding some new ones.

In this section we will give an overview of the attacks that are identified in the literature so far. We will present their relations with threats that they will necessarily cause, and those that they potentially can bring. Several threat models can be used for threat classification. Some of them are limited only to specific threats, e.g. LINDDUN (Linking, Identifying, Non-repudiation, Detecting, Data disclosure, Unawareness, Non-compliance) [35], which is focused only on privacy threats. Thus, it is not sufficient for comprehensive analysis. Models which are based on risk assessment, e.g. DREAD (Damage, Reproducibility, Exploitability, Affected Users, Discoverability) [36], PASTA (Process for Attack Simulation and Threat Analysis) [37], OCTAVE (Operationally Critical Threat, Asset, Vulnerability Evaluation) [38] have some other drawbacks, such as subjectivity, complexity or lack of technical focus [39], [40]. In order to provide a comprehensive review of all possible threats, without prioritising any of them, we will use the STRIDE model [33]. Additionally, each threat defined by the STRIDE model is focused on a specific security attribute, defined by CIA and AAA models. Therefore, security attributes affected by each identified attack will be classified according to these two models.

STRIDE model defines following threats [33]:

- Spoofing - pretending to be someone else using some of his authentication information;
- Tampering - any unauthorized and malicious modifications, including modifications of transmitted data or entries in flow tables;
- Repudiation - denial of performing or participation in some action, while the otherwise cannot be proven;
- Information disclosure - information leakage which can be exploited by unauthorized, malicious parties;
- Denial of service - service disabling for valid user by making a device (switch) or link (between switches or between switch and the controller) unavailable;
- Elevation of privilege - unauthorized user gains higher level of privilege, and thus can behave as a legitimate user, which allows him to make various malicious actions much easier.

Each of these threats compromises one of the key security attributes: authenticity, integrity, non-repudiation (accounting), confidentiality, availability and authorization respectively. One attack can realise multiple threats, and thus compromise several security attributes. Additionally, attacks are often executed combined with others, to make more damage in the network.

Possible attacks in data planes with corresponding threats are as follows:

1) Eavesdropping/sniffing/snooping - interception of communication through which an unauthorized party can gain possession of information exchanged between devices, intentionally (i.e. in the form of a MitM attack) or unintentionally. This attack is possible on switch interfaces, links between switches, or links towards the controller. In addition to information disclosure, a possible threat is also elevation of privilege, which depends on the content of the intercepted information. Therefore, eavesdropping can potentially be the first step towards some more dangerous attacks, which can include data modification or denial of service [41].
2) Side channel attacks - using available information from the network environment (such as latency, energy consumption, electromagnetic radiation etc.) to obtain confidential data. The most common example is time-based side channel attack, which is performed by sending certain test packets and analyzing the data on measured delay (e.g. the processing delay of the control plane). In this way, an attacker can obtain information about network configuration, size of switches' flow tables, network monitoring policies and other useful information [42]. This information does not contain any authentication data, so this attack, unlike eavesdropping, cannot lead to elevation of privilege.
3) Port scanning - detection of open ports in the data plane. In addition to the aforementioned attacks, it is often used as a part of reconnaissance attacks, which include gathering information using various methods. It often represents the initial step towards some more complex attacks [43].
4) Blackhole - dropping or delaying packets, usually by malicious switch or an attacker between switches [44]. This attack causes that the packet doesn't reach its destination (on time), which violates the integrity of

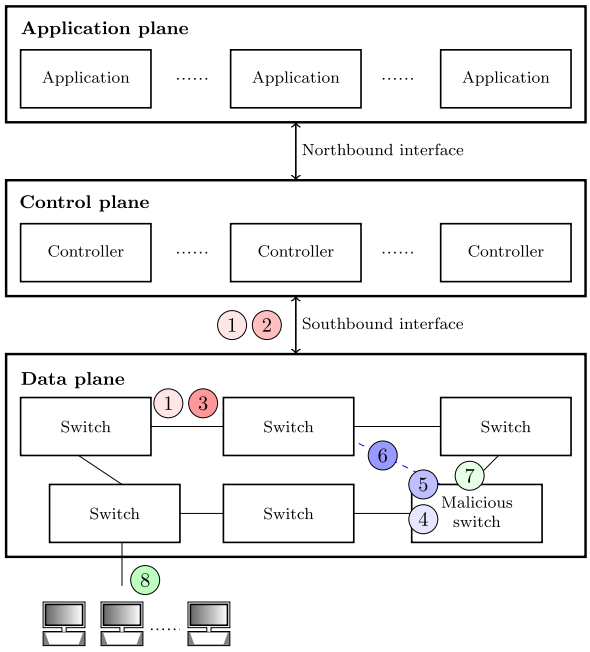


**FIGURE 4.** Possible attacks and locations where they usually occur in the SDN data plane: 1) Eavesdropping, 2) Side-channel attacks, 3) Port scanning, 4) Blackhole, 5) ARP spoofing, 6) LLDP spoofing, 7) Data modification, 8) Flooding.

**TABLE 2. Classification of security attacks in SDN data plane according to the STRIDE model: ● indicates the threats that are mandatory consequences of the attack, and ○ those that can potentially occur.**

| Attack | Threat | | | | | | Compromised security attribute | |
|---|---|---|---|---|---|---|---|---|
| | S | T | R | I | D | E | CIA | AAA |
| Eavesdropping | | | | ● | | ○ | Confidentiality | Authorization |
| Side-channel attack | | | | ● | | | Confidentiality | - |
| Port scanning | | | | ● | | | Confidentiality | - |
| *Blackhole* | | ● | | | ○ | | Integrity, Availability | - |
| ARP *spoofing/poisoning* | ● | | | ○ | ○ | | Confidentiality, Availability | Authenticity |
| LLDP *spoofing* | ● | ○ | | | | | Integrity | Authenticity |
| Data modification | | ● | | | ○ | | Integrity, Availability | - |
| Flooding | | | | | ● | | Availability | - |

the sent data. In addition, the intense packet drop or dropping a packet with specific content may cause temporary denial of service.

5) ARP (Address Resolution Protocol) spoofing/poisoning - spoofing the identity of a switch by sending fake ARP packets [22]. In this way, a malicious switch can intercept the traffic coming from the controller and gain unauthorized access to information. This type of ARP spoofing is performed as MitM attack, but it can also be performed as DoS attack, in the case of connecting one IP address to several MAC (Media Access Control) addresses.
6) LLDP (Link Layer Discovery Protocol) spoofing - sending LLDP packets with specifically modified content (e.g. different port number) to the controller (directly or through other connected devices) in order to create a non-existent link between two switches. As a result, a fake topology is created within the network [22], [42]. Therefore, these attacks are also known as topology poisoning attacks.
7) Data modification attack - any unauthorized and malicious modification, including the transmitted data, but also the forwarding rules in switches' flow tables [22]. Depending on the modifications of forwarding rules, this attack can also cause temporary denial of service.
8) Flooding - one of the most common attacks in the data plane, with the aim to make a certain switch or link unavailable for a period of time. Since its main aim is denial of service, this attack is often called "DoS attack" in the literature, although some other attacks can cause the same consequence too. Depending on the type of packets that are used, there are different versions of flooding, such as ICMP (Internet Control Message Protocol), SYN (synchronize) or UDP (User Datagram Protocol) flooding. These attacks can exhaust memory resources of a switch very quickly (TCAM (Ternary Content-Addressable Memory) exhaustion), compromise or completely block existing, valid routes, and thus disable communication with other switches [22], [45].

Fig. 4 marks locations where the previously identified attacks usually occur. Table 2 gives an overview of these attacks, with threats that they realise and security attributes that are affected by them.

For each of these attacks, it is necessary to find vulnerabilities that allow malicious parties to perform it. Some attacks are possible due to several vulnerabilities. Therefore, we will first analyse existing security solutions, since they address specific vulnerabilities related to some attack. In this way, we want to find as many vulnerabilities as possible that exist in the data plane. Security solutions overview and analysis are given in the following section.

## III. SECURITY SOLUTIONS IN DATA PLANES

Existing literature presents a number of solutions for data plane security enhancement which use various methods and implementation techniques. Most of them propose solutions for DoS/DDoS attacks, since they are the most common. Much less attention is given to attacks that compromise e.g. confidentiality, which also needs to be addressed, since lack of it can cause significant problems.

In this section, we provide a comprehensive overview of existing security solutions for SDN data planes, classified by security attributes that the corresponding attacks compromise. After that, we give an analysis of these solutions, considering methods and implementation techniques that they use. This will help to find which vulnerabilities aren't sufficiently analysed yet. Also, it will help to analyse different techniques in order to find those that can potentially be used to achieve adaptive security.

### A. OVERVIEW OF SECURITY SOLUTIONS

#### 1) CONFIDENTIALITY

Authors in [46] present a solution for mitigation of eavesdropping in SDN-based SCADA (Supervisory Control and Data Acquisition) systems. Their solution uses multipath routing technique, which sends the traffic from the same flow through several routes. In this way, it makes eavesdropping more difficult. It is implemented as an application on the controller (POX OpenFlow). Network topologies for different scenarios

are created in Mininet emulator. Experimental results have shown that in the case when there exist several routes which transmit the packets, the attacker can only partially access the data that is being transmitted. The percentage of that data depends on the scenario, i.e. the position of the attacker. Therefore, this solution provides eavesdropping mitigation. For further security improvement, multiple redundant paths should be used.

The weaknesses of multipath routing were noticed by the authors in [41], where they state that this approach is effective only if the link on the first (shortest) path is not compromised. Considering that all ACK (acknowledgement) packets are sent over the first shortest path, the attacker can block ACKs for those packets that he did not receive because they were sent over the other paths. This will lead to packet retransmission. The attacker can potentially continue to block ACKs and wait for the packet to be sent via the shortest path, thus resulting in a complete data leakage. To solve this problem, two-way multipath approach is proposed, where packets and their corresponding ACKs are sent along the same path. The experimental environment was created in Mininet, with Open vSwitch switches and POX controller. The obtained results have shown that the proposed approach provides more efficient protection against eavesdropping compared to the solution presented in [46], while the efficiency depends on the specific scenario.

In [47], authors proposed routing randomization on the granularity of packet level, using the DDPG (Deep Deterministic Policy Gradient) algorithm. DDPG algorithm is based on DRL (Deep Reinforcement Learning) and consists of two phases. The first one uses INT (In-band Network Telemetry) to collect real-time network state information. In the second phase, this data is used to generate random routing schemes, according to the QoS (Quality of Service) and security requirements. The experiment is implemented in Mininet, with P4 programmable switches. Results have shown that the proposed solution improves the protection of systems against eavesdropping attacks, with improved QoS, lower latency and higher throughput compared to other algorithms that can be used for routing randomization. Algorithms which were used for comparison are RRM (Random Routing Mutation) [48], AT-RRM (Anomaly Triggered Random Routing Mutation) [49] and SSO-RM (Spatio-temporal Stochastic Optimization Route Mutation) [50].

It can be concluded that sending packets over different paths is generally a very suitable concept for eavesdropping mitigation, confidentiality improvement and protection against information leakage. Therefore, it is often considered in literature, using different methods and algorithms. Additional improvements can be achieved by randomization of certain packet fields that can be useful to an attacker (e.g. the fields that contain an IP address or port). Such approaches can be called Moving Target Defense, whether they consider changes of route, IP address or port [47].

Authors in [51] also analyse eavesdropping in SDN data planes. They separate two levels of eavesdropping: flow entry compromised level, and switch compromised level. The proposed solution includes detection of false flow entries by query-reply mechanism and confidentiality protection using protocol fields randomization as a variant of MTD technique. Randomization includes byte level reordering and XOR (Exclusive OR) encryption. Although it enhances protection from eavesdropping, it also introduces an increase in latency of data transmission. The experiment was performed in Mininet, using a POF (Protocol Oblivious Forwarding) controller and POF switch.

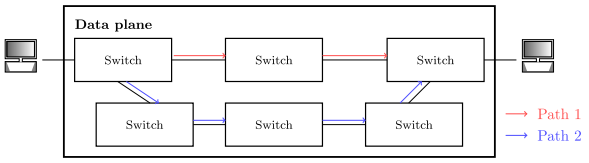


**FIGURE 5.** An example of using MTD technique in data plane: Part of traffic flow will be sent through Path 1 (expected path), and other part through Path 2, in order to improve confidentiality.

Authors in [43] have implemented SMCDS (SDN based MTD for Control and Data planes Security) framework, based on MTD to ensure protection against reconnaissance attacks, which include various techniques for collecting information. The security of the data plane is further improved by two techniques: proactive and reactive. Proactive technique uses the change of ports and IP addresses (also known as port and IP shuffling) to achieve the MTD effect. Reactive technique after detection of the attack sends the traffic to the shadow servers instead of the aimed server, selected using Round Robin or Random selection schemes. The proposed framework is implemented in Mininet, as an application on the distributed ONOS (Open Network Operating System) controller. Additionally, performance of this framework is evaluated in the terms of defender cost.

In addition to the previously described more complex routing algorithms and changing the attack target, the usage of encryption is recommended for additional confidentiality improvements and prevention of eavesdropping. In [52], [53] authors present the implementation, testing and possible application of P4NIS (P4 based Network Immune Scheme) for protection against eavesdropping attacks. P4NIS combines multipath protocols and various encryption algorithms, using data plane programmability with P4 language. It consists of three lines of defense. The first one splits all the traffic packets in different paths. The second one splits packets from one stream to different streams. The last line encrypts the payload of packets. Experimental results have shown that the implemented solution significantly complicates eavesdropping attacks, but also increases throughput compared to other implemented solutions that use single and multipath mechanisms.

Authors in [54] present dh-aes-p4, a solution which enables secure communication between data plane nodes, encrypted with AES (Advanced Encryption Standard) encryption using DH (Diffie-Hellman) keys. The solution was implemented

in Mininet, using P4 BMv2 (Behavioral Model version 2) switches. Dh-aes-p4 consists of three phases: the DH key-exchange, AES encryption and data transmission in a secure channel. DH 256-bits keys are used for AES 128/192/256 encryption, which is implemented using lookup tables in P4 nodes. Although the scenario when the controller generates private keys was also considered, it was shown that the better performance in terms of latency (both for key generation and for encryption) is obtained if the entire process is completely implemented in the data plane, on P4 switches.

Authors in [55] present a solution for prevention of MitM attacks between the controller and switches. Their solution is based on AES encryption with ECDH (Elliptic Curve Diffie-Hellman) generated keys. In this way, an eavesdropper can sniff traffic at the southbound interface, but cannot decrypt it. As expected, this solution increased delay in communication between the controller and switches. Additionally, it can be used for other types of attacks, such as flow rule insertion or flow rule tampering.

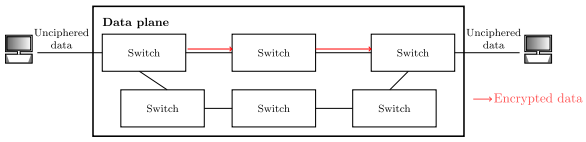


**FIGURE 6. An example of using encryption in data plane: Incoming traffic will be encrypted through P4 extern or application on the controller, in order to improve confidentiality. Encryption algorithm and length/type of encryption keys are specified in particular solution.**

For more secure communication within the data plane and the southbound interface, authors in [56] propose using IBC (Identity-Based Cryptography) [57] instead of TLS (Transport Layer Security) protocol. IBC protocol is used to establish symmetric session keys. SDN controllers act as trusted authorities, perform the function of PKG (Private Key Generator) and generate private keys for switches. Public keys are generated using the identities of network devices (e.g. MAC addresses). This change brings several advantages: simpler setup of the system, improved performance and reduced cost due to the reduced need for storage and public keys management.

Authors in [58], [59], and [60] present the implementation of a 100Gbps FPGA (Field Programmable Gate Array)-based encryptor. Their encryptor uses QKD generated keys and one of the six implemented algorithms. Algorithm can be selected using the application on the controller. In this way, confidentiality can be improved, not only due to the use of encryption, but also due to the possibility of detecting attackers in the key generation process. Using more complex algorithms like OTP (one-time pad) would provide an additional improvement.

The possibility of using QKD is also considered in [61], [62], [63], and [64]. To prevent MitM attacks, authors in [61] propose QKDFlow, based on the integration of QKD with SDN at the southbound interface to enable secure communication between controllers and switches. QKD systems should be implemented in controllers and switches that create encryption keys and enable detection of an eavesdropper. In the case when eavesdropping is not detected and a sufficient amount of key material is created, the communication between controllers is encrypted with OTP encryption.

In [62] is proposed to combine QKD with TLS (i.e. using QTLS) which enables authentication and improves confidentiality by strong encryption with QKD generated keys. In this way, MitM attacks between controllers and switches would be more difficult to perform. Combining QKD and TLS is also presented in [63], enhanced with PQC (Post-Quantum Cryptography) and classical cryptography. Such multiple-cryptography approach should provide end-to-end quantum-safe communication, securing both the data and the control plane.

Authors in [64] propose quantum layer as an additional layer to SDN architecture. This layer consists of the key generator and LKMS (Local Key Management System). The key generator runs algorithms which compute the bases for key generation protocol, used by LKMS which generate keys. LKMS also manages keys and encrypts the packets using them. The quantum layer is simulated using Qiskit, while the rest of SDN architecture is simulated using Mininet. This paper aims to show the potential of integration of QKD with SDN, to ensure secure communication even in the presence of eavesdroppers.

Authors in [65] and [66] implement MACsec and IPsec using P4. These protocols provide additional security using authentication and AES encryption at lower layers. It was shown that IPsec introduces negligible goodput decrease, i.e. that the impact of encryption in the data plane (and used BMv2 platform) is negligible. However, MACsec significantly decreases goodput, which is not a consequence of the use of encryption, but the implementation on the BMv2 platform. In addition, authors state the limitations due to which the mentioned protocols could not be implemented on the NetFPGA SUME platform.

#### *a: SUMMARY OF CONFIDENTIALITY PROTECTION SOLUTIONS*

Presented papers have shown that confidentiality protection is usually achieved through different MTD methods (e.g. as shown in Fig. 5) and encryption (e.g. as shown in Fig. 6), classical or quantum. Summary of surveyed papers that use these methods is presented in Fig. 7. Encryption can provide stronger protection, while the level of protection depends on particular implementation. Additionally, due to its basic principles, quantum cryptography (i.e. QKD as its most mature application) provides the highest level of confidentiality. Using the laws of quantum physics, QKD provides information-theoretically secure keys and enables detection of any eavesdropping attempts [63], [67] which cannot be achieved with any other method. However, in terms of data plane security, quantum cryptography is yet to be explored.

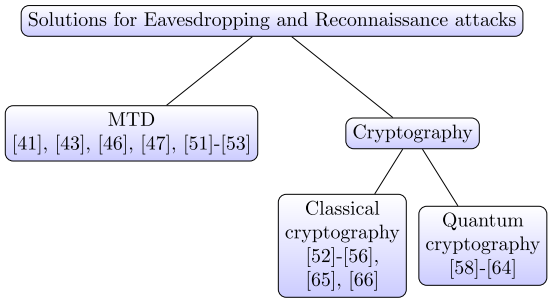


**FIGURE 7. Summary of confidentiality protection solutions: MTD and cryptography (including classical and quantum cryptography).**

### 2) AUTHENTICITY AND INTEGRITY

Spoofing attacks primarily compromise authenticity. However, identity spoofing is often performed as an initial step towards some other attacks, usually those that compromise integrity. Actually, both authenticity and integrity attacks are caused by the same vulnerability - lack of authentication. Therefore, they are analysed in this section together.

Attacks that compromise authenticity include different types of spoofing attacks. Authors in [68] present P4DDPI, which enables parsing and analysis of DNS (Domain Name System) traffic using deep packet inspection. The aim of this solution is to block malicious domains in the data plane, without involving the controller. Since the number of labels that can be parsed in one pipeline pass is limited, the implemented solution uses packet recirculation to parse all labels. It is shown that this solution has better performance in terms of throughput, delay and packet loss compared with *pfsense* [69], open-source firewall. Additionally, it doesn't occupy a significant amount of resources, which leaves space for implementation of other functionalities.

Authors in [70] are focused on spoofing attacks, including DHCP (Dynamic Host Configuration Protocol), IP and ARP spoofing. Detection of the spoofed packets is performed by checking some specific fields in their headers. Implementation was performed both in software (Mininet + BMv2) and hardware environment (NetFPGA SUME board). Experimental results have shown that the implemented solution successfully detects and drops the spoofed packets, without degradation in terms of throughput.

Authors in [71] also present a solution for detection and mitigation of ARP spoofing. This solution is implemented as an Open vSwitch extension in the data plane, in order to provide detection of attacks in real-time. It consists of several components: traffic monitor, traffic data extractor, traffic data analyzer and decision maker. The last component makes a decision whether an attack has occurred using the packet threshold as input. Another input to the decision maker is the severity level, which determines the action that this component will perform (e.g. blocking the entire switch or blocking a specific port). The experiment was performed in Mininet, with attack traffic generated with the EtterCap tool.

In [72] authors present a security mechanism which ensures switch authentication and prevention of spoofing attacks. Their solution, named BCSDN (SDN based on Blockchain), is a distributed multi-controller architecture which utilises blockchain and KSI (Keyless Signature Infrastructure). Switches in the data plane submit link information to controllers. These submissions represent transaction processes in blockchain. The consensus algorithm determines a main controller, which verifies the transaction and generates a Merkle tree. After that, the main controller generates the root hash value and issues a signature to each switch. KSI ensures that these signatures can't be forged. Therefore, communication between a controller and a switch is authenticated using this scheme.

In [73] is proposed an authentication mechanism performed entirely in the data plane. The implemented solution is based on port knocking, which means that a node needs to knock the entire predefined port sequence in correct order for successful authentication. The experiment was performed in Mininet, using BMv2 switch. It was shown that this solution doesn't affect throughput, and doesn't increase cost and complexity. This solution can also be used for mitigation of port scanning. However, it has a significant drawback: it is possible for an attacker to perform a memory saturation attack, if he knows the port sequence (which he could have obtained by eavesdropping).

Most of the existing solutions related to integrity protection are based on hash functions. In [74] two authentication schemes are proposed, implemented by additional entities at the southbound interface and in the control plane. These entities manage tables with hash values and IDs of all devices. The solution was implemented in Mininet using the ODL (OpenDaylight) controller, and additional entities were modeled using OpenHIP. Experimental results have shown better performance compared to two security frameworks discussed in the paper (*TFSv1* [75] and *SDSecurity* [76]), for different intensities and types of attacks. Additionally, considering the obtained performance under IP spoofing, MitM attack and replay attack, it can be concluded that this scheme can meet the requirement of commercial use.

In [77] authors present a P4 extension for cryptographic hashes and analyse it for three different P4 platforms: CPU (Central Processing Unit), NPU (Neural Processing Unit) and FPGA. Different hash functions were considered, because there is no single function that would provide satisfied performance for all considered platforms. Performances that are important for analysis, and differ on these platforms are latency, throughput, but also programmability. Using this extension, it is possible to achieve data integrity protection.

In [78] is presented another solution which deals with integrity violation in the data plane, covering the possibilities of modifying, adding and removing entries from switches' flow tables by unauthorized parties. It uses a Cookie field from the OpenFlow protocol to store hash values (SHAKE256) of each flow entry. Additionally, it calculates the hash value for the entire flow table, to conclude whether the switch is compromised. It consists of five components, implemented in Java, on the Floodlight controller.

Experimental environment was created in Mininet, in order to simulate different sizes of network topology. This solution is in general fast and lightweight, since it uses logical operators instead of complex arithmetic, and provides efficient detection of modified flow entries.

Authors in [79] also focus on integrity threats, considering blackhole and data modification attacks. They describe techniques for locating compromised switches that can trigger these attacks. Implementation was done on the Ryu controller and Open vSwitch switches. For localization, it is possible to send test packets to verify the consistency of the rules with the controller's knowledge, or to check the flow statistics. In addition, encryption of the packet's header or payload ensures a certain level of security, and continuous network operation even in the presence of certain malicious switches. Experimental results have shown that the detection time of a compromised switch is proportional to the number of sent test packets, which means that it depends on the topology size (number of switches, links and forwarding rules).

Authors in [42] propose FlowKeeper, a framework that effectively reduces topology poisoning attacks. FlowKeeper consists of two modules: a traffic agent and a global view agent. A traffic agent is placed between the control and the data plane and performs certain controller's functions without the need for switches to communicate directly with the controller. A global view agent is executed as an application on the controller. It enables network monitoring and provides global information to the traffic agent. FlowKeeper monitors changes of the network topology with the global view agent, and identifies the type of neighboring devices of each switch using the traffic agent. Depending on the port type from which LLDP packets are received, devices can be classified as switch, host or untested. The packets received from the latter two port types are filtered at the global view agent.

SECAP (Security-Aware Programmable) switch, another solution for topology poisoning attacks, is presented in [80]. SECAP is implemented using P4. It performs source address verification to prevent identity spoofing, and further performing topology poisoning. Source address verification consists of three phases: MAC and IP verification, ARP verification and LLDP address verification. Each switch port of SECAP is connected to a pair of P4 registers, which store MAC and IP addresses of the attached host. After learning these addresses, the port will become locked to them and drop all traffic that arrives from different addresses. In ARP verification phase, learned MAC and IP addresses will be compared with those stored in ARP header fields. If there exists any mismatch between these addresses, the ARP message will be dropped. The last phase, LLDP address verification, compares the stored MAC address with the Chassis ID field in the LLDP header and drops the message if they don't match. However, there exist some scenarios in which the described source address verification process will be successful, but attacks still can be performed. In these cases SECAP performs detection of anomalies in link characteristics, in order to detect attacks. The experimental results have shown that SECAP provides defense against topology poisoning attacks with minimal memory and processing overhead.

Different types of attacks, and also various network topologies are considered in [44]. An analysis was performed to see if the implemented solution can detect the selected attacks. The implemented solution named WedgeTail consists of two parts: detection engine and response engine. The detection engine listens to the messages between the control and data plane and determines expected paths of the packets. The response engine is executed as an application on the controller and decides on the application of management policies. Performed experiments included blackhole and ARP poisoning attacks. The results have shown that blackhole attack could neither be detected nor prevented, while ARP poisoning could be detected and stopped using monitoring. These attacks were also analysed in [81], where authors implemented an application on the controller named SPHINX, based on flow graphs which enables approximation of actual network operations and validation of all network updates in real time. Thus, it provides detection of attacks such as ARP poisoning, LLDP spoofing, and also blackhole attacks.

#### *a: SUMMARY OF AUTHENTICITY AND INTEGRITY PROTECTION SOLUTIONS*

Authenticity compromising is usually performed with the aim to compromise other security attributes, primarily integrity and availability. Presented solutions use some popular methods, like packet inspection, port knocking and hash functions to perform identity check and also protect integrity (Fig. 8). Additionally, some of them perform monitoring and classification, often realised through an additional entity, which is a popular method for availability enhancement. Therefore, these solutions can also be used for mitigation of DoS attacks, which is discussed in the following section.

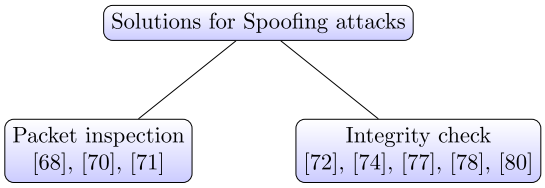


**FIGURE 8. Summary of authenticity and integrity protection solutions: Packet inspection and integrity check (based on hash functions).**

### 3) AVAILABILITY

Attacks that compromise availability in data plane are flooding and TCAM exhaustion, i.e. DoS attacks. Solutions for these attacks are usually based on monitoring and classification of incoming packets, depending on their frequency or some extracted features, using statistical or AI methods.

In order to mitigate DoS attacks, the traffic agent in FlowKeeper [42] performs the classification of the received flows based on two frequency thresholds. Flows that do not appear often (with a frequency lower than the first threshold)

will be classified as malicious and therefore will be filtered. Flows with high frequency (higher than the other threshold) will be processed regularly. Other flows will be forwarded to the global view agent for further analysis. Using this solution, bandwidth consumption by DoS attacks will be reduced from 80% to 9%.

WedgeTail [44] for the detection of TCAM exhaustion and DoS attacks caused by compromised switches uses algorithms based on expected paths. SPHINX [81] fills the flow graphs with metadata from FLOW_MOD messages used to create flow paths, in order to calculate rate of streams installation. If the installation speed is high over time, a TCAM exhaustion attack is detected. In a similar way, by monitoring metadata and bandwidth on network links, other types of DoS attacks can also be detected.

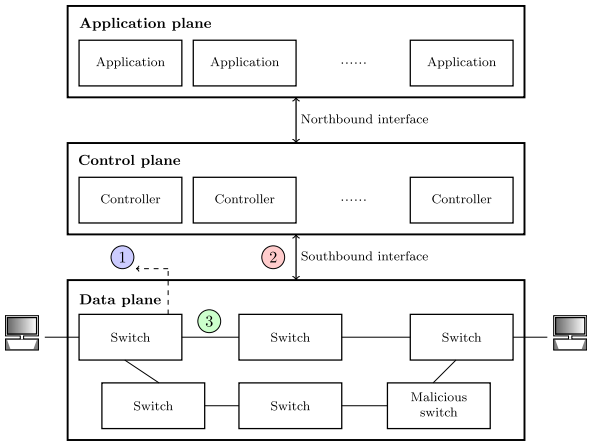


**FIGURE 9.** An example of solution for DoS attack in data plane: Malicious traffic will be dropped (1), suspicious traffic will be sent to the controller for further analysis (2), and regular traffic will be forwarded according to the rules in the flow table (3). Analysis of incoming traffic should be based on statistics or extracted features. Detection should be done using an application on the controller, or directly at the switch.

Authors in [82] create a module for classification of incoming packets, in order to distinguish benign packets from SYN flood attacks. The module is implemented in P4 language and BMv2 software switch in Mininet emulator. Classification is performed using SYN cookie technique with TCP (Transmission Control Protocol) reset method, which enables client authentication using TCP SYN cookie. In this way, only validated TCP connections will be sent to the controller for writing new flow rules. This is different compared to a normal switch that sends all SYN requests to the controller which can lead to the flow table overload.

The same approach can be also used for the attacks that are based on some other protocols, using different SYN cookie methods, like the one used in [83]. DDoS mitigation in this paper is based on client authentication which is entirely performed in the data plane, without any need for communication with the controller. It is implemented as an additional switch functionality, tested in several environments, and performed in two phases. The first one uses only one switch as a spoofed SYN flood mitigator. The second one distributes the implemented functionality over several switches, in order to adapt to the attack intensity. It was shown that this solution efficiently protects SDN switches from DDoS attacks, introducing negligible latency.

Authors in [84] compared SYN cookie with SYN authentication technique for defense from SYN flood attacks. Both techniques perform equally well, but SYN authentication is simpler for implementation.

In addition to [79], another mechanism for detection of compromised switches is proposed in [85]. In this paper, compromised switches are those that do not perform packet processing as specified by the defined rules. The mechanism is implemented through two OpenFlow applications on the controller. One of them is used to detect change in forwarding, and the other to detect change in ''weights'', that is, the amount of traffic that is forwarded on the corresponding port. The results have shown that even in the case of a very small percentage of compromised switches, their detection is possible through several iterations. Given that the solution can detect changes in the amount of traffic being forwarded, it can be used to detect DoS attacks.

In [86] authors implement P4Guard, a software configurable firewall designed using the P4 language. It can be used to extract and drop suspicious flows based on parsed parts of the packets, thereby preventing DoS and modification attacks. Compared to ClickOS-based VNGuard [87], P4Guard has shown better performance in terms of packet processing and network latency. Similar principles were used in [88] where an L3 firewall was implemented to enable DoS detection, filtering of packets depending on port and protocol, and headers parsing at different levels and making decisions based on them.

Authors in [89] present the framework which performs detection and mitigation of flow table modification attacks which aim to escalate to DoS attacks. The presented solution consists of P4 switches. Thrift port, which performs control and management operations between components of these switches, is also used to detect a malicious switch, i.e. the changed flow rules. In order to mitigate consequences of modified flow rules, authors propose fuzzy-rule-based mitigation strategy. The experiment results have shown that this technique reduces EFR (External Flow Rules) rate, and improves the overall QoS without compromising data security.

Techniques based on AI are also popular for DDoS detection and mitigation. In [90] authors present a module that extracts features on the P4 switch. Using them, this module performs very precise detection of DDoS attacks using different ML (Machine Learning) algorithms, such as RF (Random Forest), KNN (K-Nearest Neighbors), SVM (Support Vector Machine), in less than 1 millisecond.

Authors in [91] present P4-CACNN model for defense against DDoS attacks. Their solution contains two parts: Data Augmentation and Data Classification. For data augmentation, authors analyse three attention mechanisms: SENet (Squeeze-and-Excitation Networks), CBAM (Convolutional Block Attention Module) and CA (Coordinate Attention).

Experiments have shown that CA has the best performance, thus it will be used for data augmentation in further experiments. Data classification is performed using CNN (Convolutional Neural Network). After classification and detection of attack, source addresses of attackers are added to the suspicious list. Using the P4 match-action table and considering the suspicious list, appropriate actions are performed. Experimental results have shown that the proposed model achieves high accuracy in DDoS detection, with low latency due to reduced interaction with the controller.

P4DDLe proposed in [92] extracts packet-level features using programmable data planes and a counting Bloom filter. These features are essential to the ML model which utilises CNN and performs traffic classification on the controller. The chosen traffic classifier helps in achieving high accuracy in DDoS detection. Since only essential features are forwarded to the controller, the overall latency and memory overhead can be significantly reduced.

Another solution for detection of DDoS attacks based on ML is presented in [93]. The solution is implemented in Mininet, using BMv2 switches. Authors use decision tree and logic regression for classification of normal and malicious traffic. These models are transferred to the data plane in the form of flow rules, with the aim to update the data plane knowledge with models that use the most recent data in their learning process. Simulation results have shown that both models classify packets with high accuracy for six different types of attacks.

Machine learning is also used in [94], where several features are extracted from flows from switches' flow tables, and used for three different learning algorithms. The dataset is generated using Mininet, with three types of traffic (TCP, UDP and ICMP). Almost 60% of the dataset was used for training. Algorithms which were used for detection are neural network, SVM and NB (Naive Bayes). It was shown that neural network and NB have 100% accuracy, but also that neural network needs less training time.

In [95], authors have implemented a module which extracts seven features from flows in flow tables, and uses deep neural network, trained on the specified dataset. The implemented module has high accuracy, but low precision, which means that it will potentially generate false alarms. However, this can be improved by extracting more features.

Authors in [96] present a solution for DDoS based on a deep learning method which uses AE (Autoencoder) with BGRU (Bidirectional Gated Recurrent Unit). This method extracts several features from statistics and traffic flow headers, which will be used as input to the classifier to detect attack packets. Authors use Python DL (Deep Learning) library TensorFlow to train the AE-BGRU model, Scapy for packet generation and Mininet for simulation of network topology with POX controller. Experiments have shown that the proposed model can accurately detect DDoS attacks. Additionally, it can mitigate these attacks by decreasing trust values of senders and blocking them. Remarkable performance of this solution shows its potential for real-world deployment and practical applications.

Authors in [97] have implemented FloodDefender, consisting of a detection and mitigation module on the controller. These modules are implemented as Python applications on the Ryu controller. The detection module consists of three components: table-miss engineering, packet filter and flow table management component. Table-miss engineering component activates in the case of attack detection. Packet filter component identifies the attack traffic, extracting four features and using SVM classifier. Flow table management component installs monitoring rules at the attacked switch. Experimental results have shown that FloodDefender can precisely identify attack traffic. Compared to previous work, it also achieves better performance in terms of delay, packet loss and scalability.

Authors in [98] extracts six features from the flow table. They use back propagation neural networks for creating a classifier, which will detect two types of DoS attacks: TCAM exhaustion on switches and on links between switches and controllers. Their neural network consists of six neurons in input layer, representing each extracted feature, and three neurons in output layer, which represent three types of traffic which can be classified: normal, M-DoS (DoS attack with multiple flow entries) and S-DoS (DoS attack with a single well-designed entry). BP (Back Propagation) model was also compared with other classifiers, and it was shown that it is optimal.

Authors in [99] used the following models of neural networks for data plane security situation prediction: LSTM (Long Short-Term Memory), BiLSTM (Bidirectional LSTM), NDPSO-LSTM (Nonlinear Dynamic Particle Swarm Optimization LSTM) and NDPSO-BiLSTM. The aim is to act proactively in order to reduce the damage that can occur in the network due to different attacks and activities, i.e. different security situations. It was shown that NDPSO-BiLSTM has the highest accuracy and is the most suitable for this type of prediction.

Authors in [100] analyse LOFT (Low-Rate Flow Table Overflow) attack, a DoS attack with minimum necessary intensity which will lead to the flooding of flow tables. This type of DoS attack is especially dangerous, since it is more difficult to detect compared to regular DoS attacks. The LOFT attack consists of probing phase and attack phase. In the probing phase, different test packets are sent with the aim of determining the existing rules in the flow table. In the attack phase specific packets are sent at the minimum required rate, considering the information collected in the previous phase. Although the focus of this paper is on the design of a specific attack, potential countermeasures are also proposed: disrupting the first phase by generating artificial jitter or dynamically changing the timeout value, and avoiding attacks by monitoring, identifying and then removing a suspicious rule in the flow table (e.g. a rule that is always in the table, but forwards very small number of packets per second).

LOFT attacks are also analysed in papers [101], [102], [103], [104], [105], [106], [107], [108]. Authors in [101] present LOFTGuard, a solution which protects SDN against LOFT attacks introducing a small overhead. This solution maintains flow rules based on their liveness and ratio. Therefore, it can prevent TCAM exhaustion without even knowing the exact flow rules. Liveness threshold is not dynamically changed according to the real-time measurement on inter-packet delays, because OpenFlow switches cannot measure inter-packet delays in real time. Experiments have shown that the proposed solution provides effective defense against LOFT attacks, achieving good performance in terms of throughput, latency and resource usage.

LtRFT [102] is a solution for LOFT attacks based on LtR (Learning-to-Rank). It consists of three modules: monitor, ranker and mitigator. Monitor tracks the sum of flow entries and activates other modules if this sum exceeds the predefined threshold. Ranker ranks all flow entries using the features extracted with LtR model. Seven features are extracted for training the LtR model using a pairwise method, focused on relative ranking of attack flow entries. The ranking score represents the probability that the flow entry is malicious. Thus, a flow with higher ranking score will be evicted earlier. The last module, mitigator, deletes flow entries determined by ranker. This way, it performs attack mitigation since the flow table space is freed up. The experimental results have shown high accuracy in ranking normal and attack entries, low overhead and rapid response. Due to obtained performance and deployment on a real-world topology, it can be concluded that LtRFT is practicable in SDN deployments.

A solution named FTOP [103] analyse LOFT and FC (Flash Crowds) attacks, which occur when legitimate users access the service within a short period of time, resulting in reduced availability. FTOP utilises Kalman filtering to predict the flow count and two RF classifiers for attack detection. It is shown that FTOP identifies malicious rules with high accuracy, and that the used classifier outperforms five other ML algorithms (Adaptive Boosting, Gradient Boosting, Extreme Gradient Boosting, SVM and KNN) in terms of classification performance. Resource consumption of FTOP is acceptable and average latency is 2.297 seconds. In the future, this solution needs to be analysed in large-scale networks, with more complex topologies and larger amounts of traffic.

Another solution for monitoring, detection and mitigation of LOFT attacks based on ML is presented in [104]. Six features which can be used to distinct normal and attack flows are duration time, number of bytes, number of packets, mean packet size, mean packet interarrival time and mean transmission speed. These features are more widely distributed for normal flow rules. Classification is performed using XGBoost (Extreme Gradient Boosting). Experimental results have shown that FTMaster can accurately detect and mitigate LOFT attacks with low latency and slight increase of CPU usage. However, memory usage needs to be reduced.

SFTO-Guard presented in [105] consists of three modules: rule prediction module, attack detection module and attack mitigation module. The rule prediction module utilises LRCN (Long-term Recurrental Convolutional Network) model, based on CNN and LSTM, to predict changes in the rule number. Attack detection module extracts following set of features for each flow to distinguish normal and malicious flow entries: cumulative bytes, cumulative packets, average packet interarrival time, average packet size and cumulative duration. LightGBM is used as an attack detection algorithm. If LightGBM calculates that the attack probability is greater than 0.5, the attack mitigation module will be activated. This module identifies malicious flows and deletes them. The implemented solution achieves accurate and real-time detection and mitigation of slow-rate flow table overflow attacks.

FTODefender presented in [106] detects and mitigates LOFT attacks using several methodologies: EDR (Euclidean Distance Ratio), CRITIC (Criteria Importance through Intercritera Correlation) method, LightGBM (Light Gradient Boosting Machine) algorithm and LR (Logistic Regression) algorithm. EDR and CRITIC are used in the detection phase, for calculation of detection features and scores. LightGBM and LR are combined and used as a classification model for flow entry identification. After identification of malicious flows, they are added to the eviction list. Experimental results have shown that this solution effectively detects and mitigates LOFT attacks with low system overhead. Therefore, these results show the potential of FTODefender for real-world deployment.

Authors in [107] present FloRa, a solution which also utilises ML to detect LOFT attacks, but with extraction of several new features: PAF (Packet Arrival Frequency), CRS (Content Relevance Score) and PSI (Possible Spoofed IP). A powerful gradient-boosting technique Cat Boost is used as the attack detection algorithm. This algorithm is suitable for large-scale and real-world datasets, since it reduces computational overhead. It achieves detection accuracy greater than 99.4%, which is better than some previously described solutions [104], [105]. Also, it reduces CPU and memory usage.

FTSheild [108] use ensembled ML model which combines SVM and RF. It applies SVM model on selected features to classify traffic as normal or malicious. Output of the trained SVM model is then used to train an RF model. These models are deployed on the programmable data plane. This way, FTSheild achieves high accuracy, line-rate packet processing and less resource consumption.

Authors in [109] propose ALFO-Guard, a solution for adaptive LOFT attacks. These attacks have different features under different attack modes, thus the existing detection methods for LOFT attacks are not applicable for them. The proposed solution utilises GNN (Graph Neural Networks) to perform graph anomaly detection and identification of malicious flows. Flow features and structural information need to be extracted for all flows in order to detect attacks.

After the detection, malicious flow entries are deleted and added to the blocklist if the deletion count is greater than a threshold. It is shown that the proposed solution effectively detects and mitigates ALFO attacks. It is also more robust and less susceptible to environmental influences compared to some existing solutions [101], [104]. In future work, the focus should be on reducing the overhead and real-world deployment.

Authors in [110] introduce POA (Preemptive Overflow Attack) attacks and propose a solution for them. These attacks target the flow entry eviction mechanism with the aim to preempt the flow entries of normal applications. Authors propose the solution which utilises table segmentation, a score-based eviction algorithm and monitoring the continuous drift of flow feature models for defense against POA attacks. The obtained results have shown that the proposed solution is effective and has acceptable overhead. However, CPU and memory utilization increase almost linearly with both number of switches and number of flow entries. Such linear topology used in these experiments is impractical for real-world deployment.

Methods based on entropy, i.e. randomness of the packet attributes, are also very popular for detecting DoS attacks. Authors in [111] consider TCP SYN flood attacks, where the destination IP address in most packets is usually the victim's IP address. Therefore, randomness of it can be used to identify DoS attacks. The experiment was performed in Mininet, where the POX controller was modified to collect IP addresses of new packets and calculate the entropy. Normal and malicious traffic were generated using *scapy* and *hping*. Normal traffic was used to calculate entropy for detection threshold and malicious to analyse the performance of the proposed solution. It was shown that detection time highly depends on the selected size of the observation window.

In [112], the same authors as in [111] compared chi-square technique with the previously described entropy technique. The results have shown that entropy has better accuracy, but also slower detection time since it requires a larger amount of packets in the observation window to detect traffic changes. The main advantage of these solutions is their flexibility, since they can use different detection fields, thresholds and sizes of observation windows.

Authors in [113] use LogLog algorithm for estimation of number of distinct flows in the network, which provides good accuracy and small memory usage. They use normalized entropy for DDoS detection, considering volumetric attacks such as UDP flooding and DNS amplification attacks. The detection threshold is, as in the previous two papers, determined using normal traffic. It is shown that the implemented functions show equal, or better performance and accuracy compared to the similar solutions. Since all parts of the solution are implemented completely in the data plane using P4, communication overhead between switch and the controller is minimal - the switch only informs the controller when DDoS is detected.

Authors in [114] also implemented a solution for DDoS mitigation using the entropy, completely in the data plane using P4. Their solution uses an adaptive pushback mechanism, based on the following principle: after DDoS detection on a switch, it informs its neighbor intermediate switches about it through a list of suspicious IP addresses. The neighbor switch then starts its own detection, in order to conclude whether malicious traffic passes through it. If it does, it starts with filtering the packets and notifying its upstream switches. The implemented solution is very accurate, has low latency and requires a very small amount of switch's memory, which allows implementation of other applications on the same switch too.

GRAPH4 is a solution presented in [115] which also utilises entropy for anomaly detection. The control plane is responsible for generation of attack graphs, which determine nodes in the data plane that lead to a vulnerable host. In this way, entropy calculation can be performed only on these nodes instead of all nodes in the data plane, which significantly reduces the overall network overhead. The selected nodes in the data plane calculate entropy values and alert the controller if an abnormal entropy level is detected. The controller performs further actions (e.g. blocking the malicious traffic). The experiment was performed using Mininet and P4 (BMv2) switch. The results confirm that the total overhead generated by entropy calculation will be directly proportional with the number of switches that calculate the metric values. Also, there is only a small amount of additional data transferred between the control and the data plane, since this solution sends only a small alert in the case of anomaly detection.

Authors in [116] propose FlowStalker, a solution for traffic flow monitoring which consists of two phases. The first one detects target flows, depending on the predefined thresholds related to specific applications and network traffic. The second one extracts per-flow and per-packet data from target flows and thus gathers data from switches that are logically divided into clusters and linked using connections whose weights can be defined according to the specific metric which will give the best results. This solution provides accurate monitoring with small communication overhead.

A detection mechanism proposed in [117] is based on localization and statistical analysis of certain header fields. It is assumed that an attacker will perform a DDoS attack by changing some header fields, but not all of them. The proposed solution uses hash table, with header fields as columns and their hash values stored in rows, which enables fast detection of all changes of header fields. Detection is performed at the control plane, and identified malicious flows are then dropped at the data plane. It was shown that the proposed solution reliably detects DDoS attacks.

Statistical analysis is also performed in [118], in order to detect and mitigate DDoS attacks. The proposed solution StateFit consists of the application on the controller, which analyses traffic, determines rules and installs them on P4 switches, and interpreter on every switch which parses traffic

and triggers corresponding actions based on received rules. Analysis of the proposed solution was performed considering ICMP flooding attack. It is shown that StateFit successfully prevents this attack, but with significant latency which can be critical in terms of security.

In [119] is proposed a solution which separates regular and malicious traffic. This solution is based on combining statistical analysis of the number of incoming packets and analysis of selected features for different flows in order to identify malicious traffic more precisely. Also, it is based on offloading some packets to the neighbor switch, in order to save bandwidth of the attacked switch. The implemented solution precisely identifies malicious traffic and brings improvements in terms of latency, packet loss and scalability.

In [120] authors propose a firewall based on hybrid FPGA/CPU architecture. It can mitigate DoS attacks by filtering DoS traffic on the input or output interface, and by redirecting it to a honeypot. UDP flooding was considered, and detection was performed by comparing increments of UDP and returning ICMP flows counters. After detection, depending on the startup configuration, the firewall can use one of the following strategies: DROP (which installs a new rule in the flow table to drop malicious traffic before the output interface), TAKEDOWN (which reconfigures the input interface to filter malicious traffic) and REDIRECT (which installs a new rule in the flow table for redirecting). DROP and REDIRECT strategies can be easily implemented in SDN switches, and TAKEDOWN requires deep network programmability, i.e. full programmability of data plane processes below the level of flow table configuration. However, TAKEDOWN strategy shows better performance in terms of throughput and latency. Therefore, it is justified to consider the usage of deep network programmability in terms of security.

Authors in [121] propose an algorithm for detection and mitigation of DoS attacks in both control and data plane. It uses predefined thresholds to detect suspicious traffic, and after that detects whether the malicious traffic is from a host or a switch and temporarily blocks it. The solution is implemented as an application on the controller, and tested in a hardware testbed. The experimental results have shown that TCP bandwidth is consistent with the implemented solution, while otherwise it decreases in case of DoS attack.

Authors in [122] present BPP (Blockchain-based Packet Parser) realised on the switch in the data plane. It has the ability to control and check the incoming packets, including detection of blockchain header, which contains all information that is required for validity check of a block. When BPP detects malicious behavior on a switch, it informs the controller, and neighbor BPPs through P2P (peer-to-peer) connections, which continue this process through the entire network. In this way, it is possible to detect DoS attacks, and some other patterns that indicate malicious behavior.

Authors in [123] also propose a solution based on blockchain, which addresses MitM attacks at the southbound interface. Their proposed framework has an additional, blockchain layer which stores all transactions as a block in a blockchain ledger. The data stored in this layer is verification data, thus it solves issues like unauthorized access, flooding and DoS attacks in general.

#### *a: SUMMARY OF AVAILABILITY PROTECTION SOLUTIONS*

Methods used in presented solutions for availability protection can be categorised into two main groups: statistical (which can include packet inspection) and AI-based methods (Fig. 10). Some other methods, such as SYN cookie or entropy-based methods are also used. Accuracy of statistical methods highly depends on the selected threshold. AI-based methods are usually more accurate, depending on the used dataset and algorithm. In current literature, some algorithms (RF, SVM, KNN) are more used and analysed. Considering the rapid development of AI, it is expected that other algorithms will also be explored more detailed. It is also important to emphasize that performance of AI models highly depends on quality and quantity of training datasets. However, most of the datasets that are currently available are collected from non-SDN networks [124]. Thus, an increase in the number of datasets with SDN traffic will help in further development of AI-based methods for security enhancement. Additionally, an increase in the number of datasets with different types of malicious traffic or even dedicated testbeds [125] will also significantly contribute to the improvement of AI-based methods.

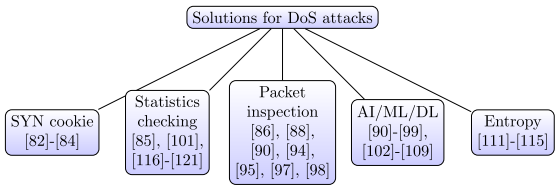


**FIGURE 10. Summary of availability protection solutions: SYN cookie, statistics checking, packet inspection, AI methods and entropy-based methods.**

### *B. ANALYSIS OF THE PROPOSED SOLUTIONS*

Considering the previous presented security solutions, it can be concluded that attacks in the data plane are possible due to the following vulnerabilities:

A) Lack of encryption - lack of encryption on links between switches and at the southbound interface enables attacks like eavesdropping and reconnaissance which lead to information disclosure.
B) Standard routing algorithms and open ports - solutions based on MTD show that lack of encryption is not only vulnerability that causes information disclosure. If the traffic is transmitted always through the same, expected paths, attackers can easily eavesdrop it.
C) Correlation between sensitive data and system information - this vulnerability leads to information disclosure caused by side-channel attacks. The usual way to mitigate these attacks is to bring some noise in the

system, to reduce this correlation. However, in the existing literature solutions for this vulnerability are not significantly analysed.

D) Lack of authentication - lack of authentication of all communication entities in SDN (controller, switches, end hosts) can cause various spoofing and modification attacks and compromise confidentiality, integrity, and even availability.

E) Limited memory resources - OpenFlow switches have limited storage capacities, hence they are often targets of DoS attacks (TCAM exhaustion).

F) Lack of intrusion detection and prevention systems - lack of a mechanism for monitoring, detecting or preventing attacks causes that suspicious traffic can easily pass through the data plane. Therefore, attackers can easily perform DoS, spoofing and modification attacks.

Also, the presented solutions show that there are a number of security methods which enhance the security of the data plane:

- MTD - different moving target defense techniques include changes of routing paths of the packet or its significant fields, such as IP address or port. These techniques provide only partially confidentiality protection against eavesdropping and reconnaissance attacks, while the level of protection highly depends on their implementations. Additionally, these techniques can't detect the aforementioned attacks. In order to provide stronger protection of data confidentiality, MTD should be combined with encryption methods.
- Encryption - usual way to protect data confidentiality, while the security level depends on the selected encryption algorithm and the method that is used for distribution of encryption keys. In the previous section, solutions that use both classical and quantum cryptography (with focus on QKD) are analysed. Quantum cryptography currently provides the highest possible level of security since it can detect the presence of malicious eavesdroppers.
- Integrity check - solutions for data integrity protection usually include methods based on hash functions, hash tables and blockchain. Blockchain is a particularly popular solution which should be more considered in the future, since it can provide transparency and non-repudiation due to its main principles.
- Header/Payload inspection - analysis of particular packet fields can help in detecting some unusual behavior in a traffic flow. Therefore, in the state-of-the-art it is usually used for protection against DoS attacks and for identity check. Header inspection is a simpler approach, thus it should be used whenever information from the header is sufficient to provide a desired level of security. Also, it is faster compared to payload inspection. Therefore, header inspection is more often used, especially in high speed switches. However, payload inspection can provide more information and thus provide protection against more specific attacks.
- Statistical analysis - includes all methods that perform comparison with some predefined thresholds (e.g. frequencies of flows, or entropy). The main challenge in these methods is defining the thresholds, since they have to correctly recognize and filter only DoS attacks, without affecting normal traffic. They are usually calculated considering normal traffic, or defined arbitrarily. Therefore, the selected threshold highly affects the quality of results which can be obtained by solutions based on statistical analysis.
- AI-based methods - all presented solutions that use methods based on machine learning, deep learning, neural networks and similar can be classified under AI-based methods. They were mostly used in DoS detection, with features extraction, and also combined with MTD techniques to achieve various randomizations. Quality of solutions which use these methods highly depends on the quality of datasets and algorithms that were used to train ML models.

Some other techniques, like digital certificates, are a mandatory part of TLS protocol which is recommended to use in OpenFlow. Thus, they are not stated and analysed in particular. Similarly, techniques that are used in one or very few number of papers (like port knocking, or flow graphs) are also not mentioned in this classification.

When it comes to the implementation techniques, two most common in surveyed papers are:

- Application/Module on the controller - besides applications that define forwarding rules, an application with desired security policy can also be defined on the controller. These applications are usually specific for the type of the controller on which they are implemented, and require some modifications to work on other types. Since these solutions require communication of the attacked switch with the controller, they introduce latency which can be significant and not tolerable in some services.
- Programmable data plane - functionalities that can enhance security can be placed directly in the data plane, due to data plane programmability and flexibility. This will help in faster and more accurate reaction against the attack, since it allows modifications in function placement. The most state-of-the-art solutions use P4 language, and various types of hardware (e.g. NPU, FPGA, ASIC (Application-Specific Integrated Circuit)). Additionally, solutions based on programmable data plane can be combined with applications on the controller, in order to exploit its global view for tasks which are not latency sensitive.

In Table 3 is given an overview of analysed security solutions, with vulnerabilities that each of them addresses (denoted using the letters assigned to them in the list when they were mentioned), security methods that are used, implementation techniques and experimental environment. The last column contains some more specific information (if they are given) related to performance and limitations of each solution.

**TABLE 3.** Overview of existing security solutions: Addressed vulnerabilities, security methods, implementation techniques, experimental/testing environment, performance analysis, scalability, limitations.

| References | Vulnerability | Method | | | | | | | | | | Implementation | | | Experimental environment | Performance analysis, trade-offs, scalability and limitations |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | MTD | Classical cryptography | Quantum cryptography | Integrity check | Parsing/Packet inspection | Statistics checking | Entropy | AI/ML/DL | SYN cookie | | Framework/Scheme | App/Module on the controller | Programmable data plane | | |
| [46] | B | • | | | | | | | | | | | • | | Mininet (POX controller) | Eavesdropping mitigation. Slight increase of packet loss ratio (can be improved with degradation in confidentiality level). Lower processing overhead compared to default POX behavior. Trade-off between confidentiality level and scalability. |
| [41] | B | • | | | | | | | | | | | • | | Mininet (POX controller) | Eavesdropping mitigation. Compared to [46]: higher confidentiality level, increased number of installed flow rules, higher average traffic. Percentage of traffic retransmission depends on the network topology. |
| [47] | B | • | | | | | | | • | | | | | • | Mininet + P4 | Eavesdropping mitigation. Improved QoS in terms of overall network delay (downward trend with defense period) and throughput (upward trend with defense period). Improved time efficiency to generate a random routing scheme. Scalability will be analysed in future work. |
| [51] | B, D | • | | | • | | | | | | | | • | | Mininet (POF controller) | Eavesdropping mitigation with detection of forged flow entries. Increased transmission delay (trade-off between randomization complexity and data transmission efficiency). Scalable solution. |
| [43] | B | • | | | | | | | | | | | • | | Mininet (distributed ONOS controller) | Reconnaissance mitigation. Increased reliability. Increase in defender cost. Low computational overhead. |
| [52] [53] | A, B | • | • | | | | | | | | | | | • | P4 + BMv2 | Eavesdropping mitigation. Compared to two state-of-the-art solutions: lower average RTT and higher average throughput. Lightweight computing resources. Backwards compatible and scalable design with support for various encryption algorithms. |
| [54] | A | | • | | | | | | | | | | | • | Mininet + P4 + BMv2 | Encryption time increases exponentially with key size. Increased computational cost if the keys are generated by the controller. Limitations of solution related to limitations of P4 functions. |
| [55] | A | | • | | | | | | | | | | • | | Mininet (ODL controller) | Increased delay due to implemented encryption and decryption processes. |
| [56] | A, D | | • | | • | | | | | | | | | | - | Reduced consumption of bandwidth and computing resources. Lower power consumption. Reduced key setup time. Improved scalability. |
| [58] [59] [60] | A | | | • | | | | | | | | | • | | FPGA (Xilinx VCU108) | Fast reconfiguration time for different algorithms. Low power consumption, high goodput. Introduced negligible latency. High key consumption rate (for QKD scenario). |
| [61] | A | | | • | | | | | | | | • | | | - | Detection and prevention of eavesdropping. Performance and robustness need further analysis. |

**TABLE 3.** *(Continued.)* Overview of existing security solutions: Addressed vulnerabilities, security methods, implementation techniques, experimental/testing environment, performance analysis, scalability, limitations.

| References | Vulnerability | Method | | | | | | | | | | Implementation | | | Experimental environment | Performance analysis, trade-offs, scalability and limitations |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | MTD | Classical cryptography | Quantum cryptography | Integrity check | Parsing/Packet inspection | Statistics checking | Entropy | AI/ML/DL | SYN cookie | Framework/Scheme | App/Module on the controller | Programmable data plane | | |
| [62] | A, D | | | • | | | | | | | | • | | | Ryu controller + Raspberry pi | Improved confidentiality and authentication. It can be considered safe against quantum brute force attack. Increased but acceptable latency. Throughput increased with MSS. |
| [63] | A, D | | | • | | | | | | | • | | | | - | - |
| [64] | A | | | • | | | | | | | | • | | | Mininet + Qiskit | The presented code does not work on newer Qiskit versions due to deprecated and removed libraries. |
| [65] | A | | • | | • | | | | | | | • | • | | Mininet + BMv2 + P4 | P4 externs increase the packet processing delay and RTT, and reduces TCP goodput. Limitations while attempting to implement the solution on the hardware (NetFPGA SUME). |
| [66] | A | | • | | • | | | | | | | • | • | | Mininet + BMv2 + P4 | Achieved reasonable performance in terms of latency and TCP goodput. Limitations while attempting to implement the solution on the hardware (NetFPGA SUME). |
| [68] | D | | | | | • | | | | | | | • | | P4 | Better performance compared to traditional firewall [69]. Constant throughput and delay. No packet loss. It usually occupies less than 50% of resources (hashbits, exact match, SRAM, actions). Limitation: ineffective against parsing encrypted DNS queries. |
| [70] | D | | | | | • | | | | | | | • | | Mininet + P4 + BMv2 + NetFPGA SUME | 100% malicious packet detection rate. Throughput remains unaffected and nearly constant. |
| [71] | D | | | | | • | | | | | | | • | | Mininet | Average detection duration depends on network size and congestion level. Processing time grows exponentially. Significant computation resources requirement. Scalable. Limitations: the solution is not deployed in real environment, and the used programming languages do not have the highest performance. |
| [72] | D | | | | • | | | | | | | • | | | Mininet (Floodlight controller) | Improved convergence time, scalability and throughput compared to the architecture with single controller. Increased CPU usage rate. |
| [74] | D | | | | • | | | | | | • | | | | Mininet + ODL controller + OpenHIP | Achieved disconnect rate less than 0.01% under three different attacks and for three different sizes of network. Scalable. |
| [77] | D, F | | | | • | | | | | | | | • | | P4 + CPU/NPU /FPGA | NPU achieves the highest throughput (for packets up to size 900B). CPU has the highest latency, and FPGA the lowest latency. Limitations: open-source implementations of hash functions were used, instead of more sophisticated ones (such as commercial FPGA IP cores). |
| [78] | D, F | | | | • | | | | | | | • | | | Mininet (Floodlight controller) | Trade-off between average detection time and generated network overhead. Increase in CPU utilization. Medium scalable. |

**TABLE 3.** *(Continued.)* Overview of existing security solutions: Addressed vulnerabilities, security methods, implementation techniques, experimental/testing environment, performance analysis, scalability, limitations.

| References | Vulnerability | Method | | | | | | | | | | Implementation | | | Experimental environment | Performance analysis, trade-offs, scalability and limitations |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | MTD | Classical cryptography | Quantum cryptography | Integrity check | Parsing/Packet inspection | Statistics checking | Entropy | AI/ML/DL | SYN cookie | | Framework/Scheme | App/Module on the controller | Programmable data plane | | |
| [79] | A, D, F | | • | | | | • | | | | | | • | | Mininet + Ryu controller | Trade-off between precision and amount of sent traffic. Increase in number of counter rules may cause a performance bottleneck and exhaust switches' TCAM memory. |
| [42] | F | | | | | | • | | | | | | • | | Mininet + Ryu controller + Python/C++ | More than 80% bandwidth is preserved under DoS attacks. Reduced workload of the controller. |
| [80] | D | | | | • | | | | | | | | | • | P4 + BMv2 | Increased latency and processing time. Reduced transmission speed. Defense against ARP cache poisoning attacks with minimal memory and processing overhead. |
| [44] | D, F | | | | • | • | | | | | | | • | | Mininet (Floodlight controller) + Java | High accuracy. No user perceived latencies. Stability challenges. Deployment over real-world network with focus on scalability left for future work. |
| [82] | F | | | | | | | | | • | | | | • | Mininet + P4 + BMv2 | SYN flood mitigation with improved scalability. No packet loss. Acceptable memory and processing requirements. |
| [83] | F | | | | | | | | | • | | | | • | Mininet + POX + P4 + BMv2 | The implemented solution has moderate impact on throughput. Increased latency (higher increase if an operation that performs writing to the TCP header is included). Moderate resource consumption. |
| [84] | D, F | | | | | | | | | • | | | | • | Mininet + P4 + BMv2/NPU/ NetFPGA SUME | Low latency and jitter for hardware implementations. Throughput close to line-rate. Low resource consumption in the case of using the NetGPGA SUME. Limitations: availability of suitable cryptographic hash functions. |
| [85] | F | | | | | | • | | | | | | • | | OpenFlow app. + Ryu controller | Trade-off between precision and detection time. Increasing the flow entry sampling rate will enhance the detection ratio, but with longer detection time. |
| [86] | F | | | | | • | | | | | | | | • | P4 + BMv2 | Low CPU overhead. Low roundtrip time for small packets. Fast processing time for low number of packets. |
| [88] | F | | | | | • | | | | | | | | • | P4 + Mininet | No support for parsing of application layer information. |
| [90] | F | | | | | • | | | • | | | | | • | P4 + BMv2 | High detection accuracy and low latency. |
| [91] | F | | | | | | | | • | | | | | • | Mininet + P4 + BMv2 | High detection accuracy and low latency. |
| [92] | F | | | | | | | | • | | | | | • | Mininet + P4 + BMv2 | High accuracy, reduced memory and control channel overhead. |
| [93] | F | | | | | | | | • | | | | | • | Mininet + P4 + BMv2 | High classification accuracy, no network downtime. ML model translated in a scalable way (without overloading the switch memory). |

**TABLE 3.** *(Continued.)* Overview of existing security solutions: Addressed vulnerabilities, security methods, implementation techniques, experimental/testing environment, performance analysis, scalability, limitations.

| References | Vulnerability | Method | | | | | | | | | | Implementation | | | Experimental environment | Performance analysis, trade-offs, scalability and limitations |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | MTD | Classical cryptography | Quantum cryptography | Integrity check | Parsing/Packet inspection | Statistics checking | Entropy | AI/ML/DL | SYN cookie | | Framework/Scheme | App/Module on the controller | Programmable data plane | | |
| [94] | E, F | | | | | • | | | • | | | | • | | Mininet + Ryu controller | High accuracy for all used ML algorithms. Lower training time for NB. Future work: usage of incremental learning for real-time performance. |
| [95] | E, F | | | | | • | | | • | | | | • | | Mininet + Ryu controller | Quite high accuracy, but low precision rate. Future work: Improvement of accuracy and precision by extracting more features. |
| [96] | F | | | | | | | | • | | | | • | | Mininet + POX controller | High detection accuracy of different attack types. High potential for real-world deployment and practical applications. |
| [97] | F | | | | | • | • | | • | | | | • | | Sw: Mininet + Ryu controller, Hw: Polaris xSwitch X10-24S2Q | Improved flow table utilization, time delay, packet loss rate and scalability. Limitations: The solution may drop normal packets in case of high attack rate. LOFT attacks are not considered. |
| [98] | E, F | | | | | • | | | • | | | | • | | Ryu controller + Centec V350 switches | High accuracy, real-time detection. Execution time increases with the network scale. |
| [99] | D, F | | | | | | | | • | | | | • | | Mininet + ONOS | Good prediction accuracy and stability. |
| [101] | E, F | | | | | | • | | | | | | • | | Floodlight cont. + EdgeCore AS4610-54T switch | Low average degradation of throughput. Saved more than 50% of memory resources. Introduced negligible latency for controller-switch communication. Slightly increased CPU usage with a higher attack rate. Low RAM usage. |
| [102] | E | | | | | | • | | • | | | | • | | Mininet + Ryu controller + OpenvSwitch | High effectiveness and accuracy, low latency (real-time performance), low resource requirement (slightly CPU usage increase). Limitations: interpretability and controllability (lack of unified control) which can lead to increase of forwarding delay and control link overhead. |
| [103] | E | | | | | | | | • | | | | • | | Mininet + Ryu controller + OpenvSwitch | High accuracy, acceptable system consumption, increased overall latency. Limitations: no unified control logic, not analysed for large-scale networks. |
| [104] | E | | | | | | • | | • | | | | • | | Mininet + Ryu controller + OpenvSwitch | High detection and mitigation accuracy. Low latency (real-time performance). Acceptable CPU usage. Limitations/Future work: system overhead needs to be reduced. Analysis in real-world network environment needs to be performed. |
| [105] | E | | | | | | • | | • | | | | • | | Mininet + Ryu controller + OpenvSwitch | High accuracy. Real-time performance. Low system overhead. |
| [106] | E | | | | | | • | | • | | | | • | | Mininet + Ryu controller + OpenvSwitch | High accuracy. Time required to cut off the source of the attack gradually decreases with increase of attack intensity. Minimal resource overhead. High interpretability. High potential for real-world deployment. |

**TABLE 3.** *(Continued.)* Overview of existing security solutions: Addressed vulnerabilities, security methods, implementation techniques, experimental/testing environment, performance analysis, scalability, limitations.

| References | Vulnerability | Method | | | | | | | | | | Implementation | | | Experimental environment | Performance analysis, trade-offs, scalability and limitations |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | MTD | Classical cryptography | Quantum cryptography | Integrity check | Parsing/Packet inspection | Statistics checking | Entropy | AI/ML/DL | SYN cookie | | Framework/Scheme | App/Module on the controller | Programmable data plane | | |
| [107] | E | | | | | | • | | • | | | | • | | Mininet + Ryu controller | Very high detection accuracy. Real-time performance. Reduced CPU and memory overhead. Limitations: high training time. Analysis of real-world deployments left for the future work. |
| [108] | E | | | | | | • | | • | | | | | • | Barefoot Tofino 100BF-32X + P4 | High accuracy. Low resource consumption. Reduced packet drop ratio. Line-rate packet processing. Future work: optimization of ML models and analysis of LOFT detection in large-scale networks. |
| [109] | E | | | | | | | | • | | | | • | | Mininet + Ryu controller + OpenvSwitch | High accuracy of ALFO detection under different attack modes. CPU utilization and memory usage need to be reduced. Analysis of real-world deployments left for the future work. |
| [111], [112] | F | | | | | | • | • | | | | | • | | Mininet + POX controller | High detection accuracy and flexibility. Larger packet volume can cause an increase in detection time. |
| [113] | F | | | | | | • | • | | | | | | • | Mininet + P4 + BMv2 | High detection accuracy. Memory efficient. Low communication overhead between the controller and switches. |
| [114] | F | | | | | | • | • | | | | | | • | P4 + BMv2 | High accuracy. Negligible processing delay. Low memory consumption. |
| [115] | F | | | | | | | • | | | | | | • | Mininet + P4 + BMv2 | Detection at a line rate. Reduced overall network overhead. Limitations: completeness of vulnerability enumeration. Changes in network environment are not assumed. |
| [116] | F | | | | | | • | | | | | | | • | P4 + BMv2 | Accurate monitoring with relative low overhead and delay. Scalable. |
| [117] | E, F | | | | | | • | | | | | | • | | Omnet++ | Reliable detection. Limitation: Attackers that can change all header fields simultaneously cannot be detected. |
| [118] | F | | | | | | • | | | | | | • | • | Mininet + ONOS + P4 + BMv2 + Java | Reduced latency and signalling overhead. Latency may soar in large-scale networks. Limitations: consistency issues in multi-vendor and large-scale networks. |
| [119] | E, F | | | | | • | • | | | | | | • | | Sw: Mininet + Ryu; Hw: Polaris xSwitch X10-24S2Q | Precise identification of attack traffic with low CPU utilization, low delay and low packet loss. Scalable solution. |
| [120] | F | | | | | • | • | | | | | | | • | FPGA/CPU + Java | DoS mitigation without significant degradation of latency and throughput. |
| [121] | E, F | | | | | | • | | | | | | • | | Ryu + Python + Zodiac-FX switch | No significant degradation of bandwidth (compared to the case when the network is not under the attack). Low CPU utilization. |
| [122] | D, F | | | | • | • | | | | | | | | • | P4 + FPGA | Attack detection with increased speed, lower latency and reduced hardware resources compared to other parsers. Limitations: Scalability not supported. Potential issues related to storage. |

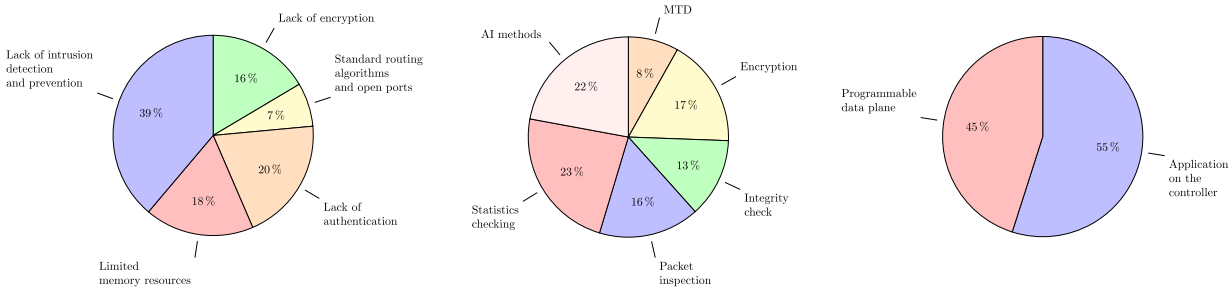


**FIGURE 11.** Statistics on the number of surveyed papers presented in Table 3 regarding addressed vulnerabilities, used security methods and implementation techniques respectively.

Fig. 11 shows statistics on the number of surveyed papers presented in Table 3, in terms of addressed vulnerabilities, used methods and implementation techniques. It can be seen that most of them analyse DoS attacks, which exploit limited memory resources and lack of intrusion detection and prevention. Much less attention is given to eavesdropping attacks which exploit lack of encryption. Also in these terms, the possibility of using quantum cryptography as currently the only way to achieve information-theoretical security is neglected. This opens up possibilities for further security enhancements of data planes, which will be presented in next sections.

## IV. ADAPTIVITY ENABLERS

In previous sections we give an overview of the potential security attacks that are observed in the literature so far, and various solutions for them. However, most of them are focused on solving one selected threat, without considering a possibility to adapt to more complex security situations in the network which can arise as a consequence of network, attacker or user dynamics. In this way, adaptivity in security can be considered in two following terms:

- Adaptivity to achieve different levels of security - different services, depending on their purpose and the data that they transmit, require different levels of security. Some of them, which include transmission of sensitive data, require the highest possible level of security, since lack of it can lead to critical repercussions. Security level for each service should be specified by quality-of-security. The system should be able to adapt to different levels of quality-of-security, and in that way meet user's requirements, i.e. respond to user dynamics. Additionally, the system should also appropriately react in cases when it can't provide the requested security level.
- Adaptivity to achieve protection against different security attacks - security attacks usually occur combined, rather than individually. Therefore, it is necessary to ensure protection against as many of them as possible as reaction to attacker dynamics. In this context, the system should recognize the type of attack, in order to adapt to it and use appropriate protection mechanism.

Based on the previous presented analysis, we have selected some proposed mechanisms which can be used to achieve adaptivity in aforementioned terms of security. These mechanisms are explained in detail in the following subsections.

### A. PACKET INSPECTION

Packet inspection is a security method which enables parsing and analysis of specific parts of the packet. Depending on the part of the packet, there are two types of packet inspection (Fig. 12):

- traditional packet inspection (or header inspection), which analyses only the header of the packet;
- deep packet inspection, which additionally considers the payload of the packet.

Traditional packet inspection includes analysis of a packet below the application layer, i.e. IP and TCP header. As it was presented in section III, this type of analysis can help in detecting spoofed IP addresses or DoS attacks. Also, with the increasing popularity of P4 language which can efficiently parse packet headers, it has become a part of security solutions for various security threats. However, analysis of only header is not sufficient in some cases.

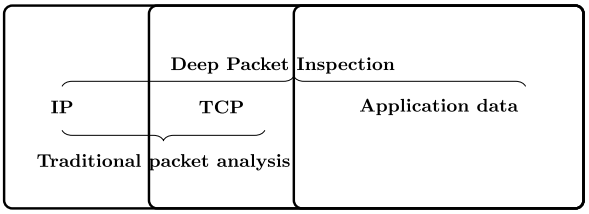


**FIGURE 12.** Packet inspection: Traditional packet analysis covers only IP and TCP headers, while deep packet inspection additionally analyses packet payload.

DPI (Deep Packet Inspection) analyses both header and payload of a packet [126], which can help in detecting specific patterns in order to enable or improve some network services such as monitoring [127], traffic classification [128] and scheduling [129], load balancing, or security.

In [130], authors propose an extended SDN architecture, including modified OpenFlow messages, CLA (Classification) and DPI modules for classification of incoming traffic, and SR (Service Routing) module which is responsible for

policies updating. The CLA module performs classification based on the packet's headers, thus can be used for detection of attacks such as SYN flooding. If the packet cannot be classified by the CLA module, then DPI module analyses its payload to find malicious patterns. Therefore, this solution can also detect web application attacks. Since the proposed solution performs traffic classification on the data plane, the traffic overhead to the controller is significantly reduced.

In [131] is presented DeepMatch, which performs deep packet inspection in the data plane using network processors. Packets are first classified according to their headers, after which some of them are forwarded to payload scanning. After that, corresponding actions according to the security policies are performed (allow, drop, redirect or rate limit). This solution drops the packets directed towards the server that match malware rules, and also prevents DoS attacks by rate-limiting. Some more complex tasks which include larger regular expression matching or reordering will reduce the throughput, but currently, this solution provides throughput larger than 10Gbps with satisfied QoS.

In [132] and [133] authors propose DPX, an extension for data planes which enables security services, one of which is DPI. It has a modular design, in order to keep simplicity of the data plane which can be easily extended on demand. Also, authors emphasize the significance of an implementation in the switch, rather than as an application on the controller. Their implementation was done in both software and hardware, using Open vSwitch and NetFPGA SUME board. The experimental results show that DPX can detect and block DoS attacks, port scanning and remote exploit.

Authors in [134] present DeeP4R, a solution which works as an application-layer firewall filtering malicious URLs (Uniform Resource Locator), using DPI and packet recirculation. For an incoming packet, DeeP4R makes its copy which will be used for checking if the target string occurs. Considering the principle of recirculation, this checking is performed byte by byte. If the target string is found in the copied packet, the original packet is dropped. The solution is implemented using the Netberg 710 (Tofino) switch, but using only standard P4 functionalities to make the code portable to other platforms. The experimental results have shown that DeeP4R successfully filters unwanted URLs, with good performance in terms of latency and throughput.

Authors in [135] analyse the possibility of using DPI for intelligent intrusion detection in software defined industrial networks. They have implemented a DPI solution on the controller, and tested it under an ICMP flooding attack. However, they didn't consider real network scenarios with complex topologies and huge amounts of traffic, which can cause scalability problems and result in the controller becoming a bottleneck.

Authors in [136] discuss the optimal number of DPIs that should be deployed in the network, and also their locations considering cost, bandwidth and violations. Their solution is based on genetic algorithm. Authors in [129] propose an algorithm which determines DPI proxy towards which the ingress traffic should be sent, in order to minimize overall latency. Challenges like these, related to network function placement and its impact on performance can be overcome by exploiting network programmability and flexibility, which is discussed in section IV-C.

Solutions based on packet inspection are often combined with machine learning algorithms, supervised [137], [138], semi-supervised [139], [140] or reinforcement learning [141]. More details related to specific techniques are given in section IV-B.

Presented papers show that packet inspection is gaining more and more popularity, and it is already used in some research which is directly related to security, or in some that can be possibly later used to enhance security, such as classification. This will allow efficient distinction between traffic flows with different security requirements. In combination with AI methods, DPI can become an important approach in achieving adaptive security.

### B. ARTIFICIAL INTELLIGENCE

Machine and deep learning are among the most popular artificial intelligent techniques used to improve network security and performance in general. In state-of-the-art literature, various solutions based on supervised, semi-supervised, unsupervised and reinforcement learning [142] are proposed and used for traffic engineering, classification and monitoring. These solutions are mostly trained on publicly available datasets which include a significant amount of specific traffic patterns, enabling them to learn how to adapt to various network situations and conditions. They also used various classifiers, which can be compared in terms of accuracy and resource consumption in order to find the most suitable one.

Authors in [137] compare different binary classifiers for inspection of encrypted and unencrypted packet payload. Out of three selected classifiers (SVM, MLP (Multilayer Perceptron) and DT (Decision Tree)) for detection of encrypted malware traffic, SVM has the lowest accuracy by far, while Decision tree shows the best classification performance. For unencrypted traffic, authors compare six classifiers (KNN, LR, DT, MultinomialNB, RF and SVM). SVM and DT have shown the highest accuracy, but also have higher training time.

In [138] authors propose signature-based traffic classification based on supervised machine learning, which helps in identification of malicious traffic and DDoS mitigation. RF and MLP were used as classifiers. In contrast with RF, MLP has shown ability to classify attacks that significantly differ from training patterns. Additionally, test runtimes for MLP are faster, but for both classifiers are negligible in general.

Authors in [143], [144] present HALIDS (Hardware-Assisted Machine Learning IDS), IDS for in-network monitoring based on machine learning. Their solution uses a programmable data plane, in order to emphasize the

importance of early-decision making at the data plane level. These programmable switches extract and calculate features for each incoming packet, process them with previously trained RF and make decisions on classifying them. If this decision cannot be made with a satisfying confidence level, it will be deferred to an oracle. Oracle is also trained with RF, but with all available data and higher model complexity. Experiments have shown that this solution provides detection of malicious flows with very high accuracy. The results confirm that HALIDS can be implemented in a real network as an efficient, adaptive security mechanism.

DDoS mitigation based on ML is also analysed in [145]. The presented solution uses both supervised and unsupervised learning. A packet that comes to the switch and does not have a corresponding flow rule in its flow table will trigger a DDoS detection module. This module will extract some features and send it to the trained ML model for classification. SMO (Sequential Minimal Optimization) algorithm was used for training and testing. If the model cannot decide whether the packet is malicious or not, it will send the packet for further analysis to the module based on unsupervised learning method. Evaluation of the proposed solution is performed in the case of its deployment in a real SDN environment. It was shown that only a small amount of malicious traffic reaches the target host.

Authors in [146] have also used supervised learning, specifically for security improvements in IoT systems. A significant amount of packets marked as malicious or normal was used to train a dilated CNN. Data plane in this case is considered as an IoT gateway, and programmable using P4 language, which enables intrusion detection inside the IoT gateway using deep learning methods.

Authors in [139] combine deep packet inspection and semi-supervised machine learning to achieve adaptive QoS traffic classification. DPI is used to associate part of the flow with specific application, forming a partially labeled dataset which will be used to train the classifier. Using a small number of labeled data, and a large amount of unlabeled data, semi-supervised learning can achieve even better learning effect compared to supervised and unsupervised learning. Experimental results on realistic network traffic data (captured in a campus network) with three classifiers have shown that the implemented solution can accurately classify incoming traffic according to required QoS. Semi-supervised learning is also used in [140], where authors present a solution for anomaly detection on the data plane and prediction of DDoS and botnet attacks. Their solution achieves better accuracy than a fully-supervised system trained on the same dataset.

Deep learning was used in [147], [148], [149], [150], [151], [152], and [153]. Authors in [148] present a deep learning based DDoS detection system, which can detect these attacks in both control and data plane. Their solution uses SAE (stacked autoencoder) based on unsupervised learning. Experimental results have shown that this approach achieves better accuracy compared to soft-max and neural network, which are actually building blocks of SAE.

Stacked autoencoder is also used in [149] and compared with MLP and CNN, in order to perform classification of encrypted network traffic. SAE and CNN have better classification performance, but MLP consumes less computational resources and provides larger bandwidth.

Deep learning for DDoS detection is combined with information entropy analysis in [150]. CNN classification algorithm shows higher accuracy compared to DNN (Deep Neural Network), PSO-BPNN (Particle Swarm Optimization-Back-Propagation Neural Network), RF and SVM, but also has higher training time. However, training can be performed offline, so high training time is not significant.

Authors in [152] also compare various deep learning techniques (CNN, LSTM, CNN-LSTM, SVC-SOM (Support Vector Classifier-Self Organizing Map), SAE-MLP) for traffic classification in order to detect DDoS attacks. SAE-MLP has the highest accuracy (99.75%) and efficiently classifies traffic into malicious or normal.

In [153] authors propose a solution which combines RNN (Recurrent Neural Network) with autoencoder, also for DoS attacks detection. Their training dataset contains 12 types of DoS attacks. However, tested traffic can be classified in one of only two categories - normal or DoS attack. The performance of the used model was tested using different learning rates. It is shown that the highest accuracy is achieved for the lowest learning rate. Additionally, in terms of accuracy, the proposed model outperforms the following six algorithms: LR, SVM, RF, Booster, DT and NB.

Another solution based on deep learning, which enables detection of DDoS attacks is proposed in [141]. This solution is based on double deep Q-network, which provides efficient traffic flow monitoring, and thus enables efficient anomaly detection.

Traffic classification based on deep learning and implemented directly in the data plane is presented in [154]. This solution uses CNN for accurate classification of similar traffic, and LSTM model for classification of remaining traffic with low overhead. Additionally, authors use a programmable data plane in order to reduce the overhead of feature storage.

Authors in [151] use hybrid deep learning to detect reconnaissance attacks. Their solution comprises LSTM and CNN in order to achieve improvements in terms of accuracy and time efficiency. It efficiently detects XSS (Cross-Site Scripting), botnet and port scan attacks.

Another solution based on deep learning is presented in [155]. This solution utilises CNN which can quickly and accurately identify malicious traffic. However, recognition speed has shown some fluctuations with time, which is possibly caused by complexity or variability of network traffic. Therefore, this model needs further improvements in order to enhance its stability for usage in practical applications.

Deep learning can also be used to secure the data plane from covert channel attacks [156]. Such attacks represent the main cause of rule conflicts in SDN. They exploit the

inconsistencies in the network isolation filters with the aim of delivering malicious traffic. In the proposed solution, Feed-Forward Back Propagation neural network is used for classification. If a packet is classified as malicious, the host that sent it will be blocked and the flow table will be updated. Experimental results have shown that the proposed solution can detect and stop covert channel attacks in real time without significant degradation of performance.

Authors in [157] design Federated learning-based Security (FedSec) strategy which utilises FL (Federated Learning) and DL for securing the MC-SDN (Multi-Controller SDN) networks from attacks in the data plane. The used network dataset NSL-KDD is enriched with SDN-specific features in order to reflect real-world network traffic. Detection of malicious traffic is performed using GCNN-GRU (Graph Convolutional Neural Network-Gated Recurrent Unit). Usage of MA-HS (Mayflies-Harmony Search) algorithm helps to optimally select the features from the dataset, thus improving intrusion detection accuracy.

ML techniques can be combined with different tools, such as SIEM (Security Information and Event Management) which collects and analyses data from different sources to perform real-time monitoring and defense against various attacks [158]. In this work, SIEM is integrated at various points of SDN architecture. In the data plane, it can help to detect unauthorised access and data leakage. ML models are trained on historical network traffic data and can be integrated in the system at different points. Several ML models were considered, including SVM, DT, RF, AE, RNN and CNN. The results have shown that RF and SVM achieves the highest accuracy. This solution shows that ML can be combined with tools that are already in use in real-world deployments. Additionally, integration of ML and SIEM provides responses to different security attacks and also helps in achieving adaptive security.

Benefits of combining artificial intelligence with programmable data planes are emphasised in several works [159], [160], [161], [162]. These solutions integrate programmable switches with different ML techniques, thus providing both high accuracy and high performance. In this way, it is possible to perform monitoring, traffic analysis or intrusion detection at line rate. This will be especially important in developing security solutions for next-generation networks.

Authors in [163] present an IPS (Intrusion Prevention System) for 6G networks based on deep learning. This IPS uses CNN for classification, RNN for traffic pattern analysis and RL for policy updates. It includes two components: a misuse detection engine which uses DT for signature-based detection of known threats, and an anomaly detection engine which uses Isolation Forest algorithm for identification of unknown threats. Experimental results have shown that the proposed IPS achieves high accuracy with efficient resource utilization, offers scalability and high-speed performance.

The potential of massive distribution of intelligence for security enhancement in next-generation networks is described in [164]. Authors propose a paradigm in which programmable data plane devices detect potential attacks and immediately block them, without any intervention of humans or the controller. This should be enabled by splitting a stronger ML model into weaker models and distributing them to programmable switches. The presented paradigm should maximise security coverage, reduce response time and offer high scalability compared to previous solutions which deployed ML models completely on network devices. In the future, it will be important for AI-based solutions in general to consider if there is a need to deploy the AI model in each switch. Placing these models only on some selected switches will have a positive impact on network latency [165].

The presented analysis shows that AI methods have already been used in a significant number of papers related to data plane security, mostly for tasks that can help in defense against DoS attacks (monitoring, classification etc.). However, in terms of adaptivity, they have not achieved their full potential yet. Additional attention should be given to security of dataset and learning models, i.e. defense against adversarial attacks which has not been explored in the related literature so far. Also, most of the research related to AI have not addressed real-world challenges and issues that can occur during integration of algorithms into existing network infrastructures, such as computational requirements and model updates [166]. Therefore, more attention should be paid to practical implementations of AI solutions in real-world deployments.

### C. DATA PLANE PROGRAMMABILITY

One of the main concepts on which software-defined networks are based and a key feature that their architecture has brought is increase in programmability. Programmability provides integration of SDN with NFV (Network Function Virtualization), a concept which allows defining various network functions through software and their execution on SDN nodes. It includes different functions, such as firewall, load balancing, monitoring, intrusion detection and similar which can be used for security enhancement [167], [168].

Since data plane programmability represents an essential factor to achieve flexibility and adaptivity in various terms, such as topology, network functions and resources [169], it is very important to consider its contribution in terms of adaptive security. Additionally, as it was shown in section III, a significant number of existing security solutions use or at least consider data plane programmability. This shows us that the significance of this concept has been already recognized in the literature.

Currently, most solutions based on programmable data plane use P4 language. It has gained popularity due to its suitable features, like reconfigurability, protocol independence and target independence [170]. Many operators, companies and switch manufacturers have already adopted P4 in their solutions [171]. Therefore, it is certain that P4 has a future in software-defined networks, and also that it will probably become one of the main pillars of data plane programming.

In [172] are presented various solutions based on P4 programmable data planes for network security enhancement. They address different security threats, but mostly those related to availability, i.e. DoS attacks. This is expected, since P4 enables efficient packet parsing, one of the most common methods related to DoS detection. On the other hand, confidentiality protection has gained less attention, mainly due to limitations of P4 language which supports only basic arithmetic operations [171], [173]. However, several proposed solutions consider a combination of programmable data planes and encryption, usually using P4 externs.

Additionally, programmable data plane solutions can be realised in both hardware and software, including CPU, NPU, FPGA and ASIC. These platforms differ in terms of flexibility, performance (throughput, jitter and latency) and limitations related to resources. Hardware solutions can achieve better throughput, jitter and latency, out of which FPGA and ASIC significantly outperform NPU. However, they are limited in terms of resources and flexibility. Also, they are less used and tested in existing solutions [172].

One of the problems which programmability can also solve is related to network function placement. Several papers [174], [175], [176] analyse methods for optimal placement of network functions. With programmability and reconfigurability, the position of a specific network function should adapt to specific situation and network conditions, including different security threats. It should be emphasized that reconfiguration will bring additional latency. Thus, it is necessary to find a way to perform reconfiguration, which will sufficiently protect the system with negligible performance degradation (e.g. using deep network programmability instead of waiting for instructions from the controller).

Considering all of the above, we can conclude that programmability will play a significant role in data plane security and achieving adaptivity. Additionally, a platform for solution development should be chosen carefully in order to achieve compromise between performance and level of security.

## V. FUTURE DIRECTIONS: TOWARDS ADAPTIVE AND SECURE DATA PLANE

In this section, we use observations presented in previous sections to provide a possible approach towards a comprehensive, powerful solution which will enhance the security of SDN data planes. This solution should be based on mechanisms presented in section IV in order to achieve adaptivity, and in that way ensure appropriate level of security for different services and react to different types of security attacks, under various network conditions. Authors in [23] emphasise the importance of AI techniques to achieve adaptivity, and present some existing combinations with MTD techniques. Our work considers combinations with DPI, different encryptions and reconfigurability which can additionally improve adaptive security. In the following sections, we present three approaches: adaptive encryption, adaptive function placement and security-aware adaptive routing which should respond to user, attacker and network dynamics respectively.

### A. ADAPTIVE ENCRYPTION

In section III it was shown that in the existing literature DoS attacks have attracted the most attention. Compared to them, eavesdropping and other attacks that compromise confidentiality are neglected and haven't been sufficiently analysed, although they can represent a significant security threat. Since these attacks are usually only the first step towards others, more complicated attacks which can wreak havoc in the entire network, it is important to ensure a high level of confidentiality. A typical way to ensure confidentiality is encryption, using classical or quantum cryptography, which can provide different levels of security.

Quantum cryptography will provide the highest possible level of security (information theoretical security), since it is based on the principles of quantum physics [177]. Thus, it will be one of the key technologies for 5G/6G networks [67] and services where lack of security can lead to critical repercussions [8], [178]. A number of practical implementations of QKD networks have already been deployed in Europe (Berlin, Madrid, Poznan etc.) [179], with interfaces and communication between them specified in ETSI QKD standards: [180], [181] for key delivery and [182] for interface between SDN-QKD node agents and the SDN controller. Also, there are many proposed architectures that integrate SDN with QKD [183], with even practical implementations of these integrations [184], [185], [186], which is not surprising due to many benefits that SDN brings (e.g. in order to integrate QKD in the network, it is easier to define new management and routing policies on the controller instead of installing a new routing protocol on every router in the network). Therefore, it is justified to consider the usage of quantum cryptography, which is still not sufficiently analysed in SDN data planes.

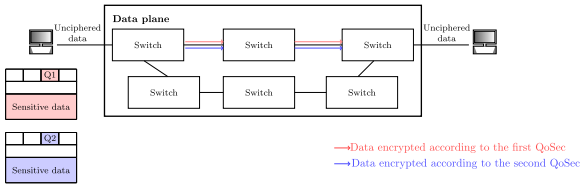


**FIGURE 13. An example of a proposed solution for adaptive encryption: Two incoming packets have different quality-of-security requirements, which can be determined through particular fields in their headers or their payloads. They are encrypted in different ways, in order to satisfy their requirements, but can be sent through the same path, depending on rules in the flow table.**

To provide the required quality of security, the following can be considered:

- encryption algorithm - e.g. OTP for higher level of security, and AES for less sensitive data;
- types of keys - e.g. QKD for services that require very high level of security, and DH for others;

- key size - which depends on the chosen algorithm and type of keys.

Authors in [139] use deep packet inspection and machine learning to classify traffic according to the required quality of service. In similar way, these two techniques can be combined to provide different levels of security:

- The required level of security, which depends on specific service, can be specified using in-band signalization, e.g. in a packet header as shown in Fig. 13. Such a metric which would define security in this way still does not exist. Therefore, we should define the quality of security for these purposes, which will indicate which type of keys and encryption algorithm should be used. Additionally, this will require a field in a packet header which should transmit such information, which means creation of a new protocol for this communication. However, this is not a problem for P4 language, since it can parse any user-defined headers [134].
- Besides the header, a payload should also be used to determine the required level of security. Using DPI, a switch can find the type of traffic and application, and thus determine the required level of security. This system should be trained using some of the mentioned ML algorithms, on dataset which include a significant amount of traffic patterns for various applications.
- Quality of security should be also defined using out-of-band signalization. In this way, protocols for key delivery such as ETSI GS QKD 004 [180] should be considered. This protocol defines some particular fields in its proposed packet format which can be used for this purpose.
- Additionally, it should be considered that the content of the packet is not completely sensitive. In this way, DPI should be used to find particular parts of the packet that contain sensitive data. These parts should be encrypted in a more secure way compared to the rest of the packet, e.g. using OTP encryption with different key sizes. In this case QKD can also be helpful, since the key length can be set to different values in user's requests, in the way that is specified in ETSI QKD standards. This solution should also be trained using some of the mentioned ML algorithms, on dataset which include data with different sensitivity.

Another important scenario that should be considered is the one when the required level of security cannot be provided due to the current situation in the network (e.g. temporary lack of required keys). Thus, the system should adapt to the particular situation and provide the highest level of security which is possible at that moment. These adaptations will depend on the entire architecture and specific network situation, but should be predicted and achieved using AI techniques. Additional possibilities for similar scenarios will be considered in section V-D.

Considering previously mentioned cases, the further solution should provide encryption which can adapt to different security requirements and data sensitivity.

### B. ADAPTIVE FUNCTION PLACEMENT

Encryption should be provided as a mandatory, built-in feature in SDN data plane devices. However, using encryption will only protect confidentiality of the transmitted data. In order to provide a strong and comprehensive security solution, threats that affect other security attributes should be considered.

As it was presented in section III, protection against DoS attacks has gained the most attention so far. Thus, it would be useful to provide a solution which will encrypt and pass the traffic until the moment when DDoS is detected. After that, malicious traffic should be blocked, dropped or redirected.

In order to achieve this, it is important to consider where DDoS-related functions can be placed. One possible way is to use dedicated middleboxes for this purpose, which will first inspect traffic to determine if incoming packets are part of DDoS attack. However, this approach will increase latency of data transmission. Also, there is a problem related to the number and placement of these middleboxes.

A better way to perform DDoS-related functions should include NFV, which can provide adaptivity and fast reaction. The most efficient ways to include DDoS detection in a security solution are as follows:

- The controller can be used to detect DDoS attack on a particular switch due to its global view. After DDoS detection, the controller can adapt and reconfigure the switch (Fig. 14), to treat malicious packets appropriately (block, drop or redirect) instead of their encryption and forwarding.
- DDoS detection can be performed directly on a switch (whether inspecting a packet header and performing statistics analysis, or inspecting a payload to find suspicious patterns). After DDoS detection, the switch can reconfigure itself to adapt to the current situation, and treat malicious packets appropriately (block, drop or redirect) instead of their encryption and forwarding. This reconfiguration can be done due to the concept of deep network programmability.

In this way, there is no need for a dedicated firewall in SDN networks, since switches in the data plane can be used for this purpose. This concept represents adaptive function placement, since a switch can adapt to a particular network situation, and depending on it act as a firewall or an encryptor. Additionally, QKD can be used to improve NFV security [187], [188], [189], thus it has the potential to be used in both adaptive encryption and adaptive function placement based on NFV.

### C. SECURITY-AWARE ADAPTIVE ROUTING

Adaptive techniques described in previous sections assume that all network devices and links between them are up and stable. However, there is always a risk that some switch or link in the network will fail, which will make certain routes unusable causing availability degradation and packet loss. Additionally, if an attack or malicious switch is detected on a certain path, traffic should be redirected to some other path,

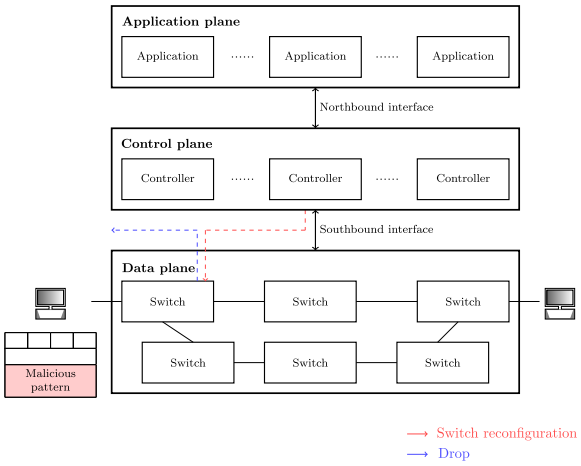


**FIGURE 14.** An example of proposed solution for adaptive function placement: A switch receives a packet with malicious pattern. The controller detects suspicious traffic and performs switch reconfiguration. Therefore, instead of encryption and forwarding, it drops the packet.

which does not suffer security problems. To achieve this, a security solution should provide adaptive routing, and in aforementioned situations redirect the traffic from primary to alternative, stable and secure path. This approach, based on sending the traffic through different paths, is known as multipath routing, and in current literature (related to SDN) is mostly explored in terms of traffic engineering, quality-of-service [190] and load balancing [191], but less in terms of security. However, similar principles can be used to perform security-aware multipath routing.

Two variants of multipath routing are proposed in current literature: proactive and reactive. Proactive method has limitations due to which it cannot be used for security-aware routing: precalculated routes can suffer a link or switch failure [192] or an attack, before it happens on a primary route. Thus, it is necessary to use a reactive approach and find a convenient path through which the traffic should be sent to its destination.

Several approaches can be used to perform multipath routing:

- The controller can be used to detect network failures due to its global view. During this process, the controller can also use machine learning for easier detection and classification of network faults based on specific patterns (e.g. deep reinforcement learning has already been considered in [192], [193], and [194], and it was shown that it can be used to achieve security-aware multipath routing). After it detects that a particular switch or link is down, it should find a new route for a traffic flow, and install a new rule in the switch's flow table (Fig. 15). The information about the new path can also be stored in a certain packet header's field, in order to reduce switch storage overhead [195].
- Failure detection and rerouting can be performed by switches in the data plane. Switches can send probe packets periodically, or in some specific case (e.g. low packet rate [196]) and in that way realise if there exists failure in the network. However, without precalculated paths, interaction with the controller is inevitable for further rerouting.

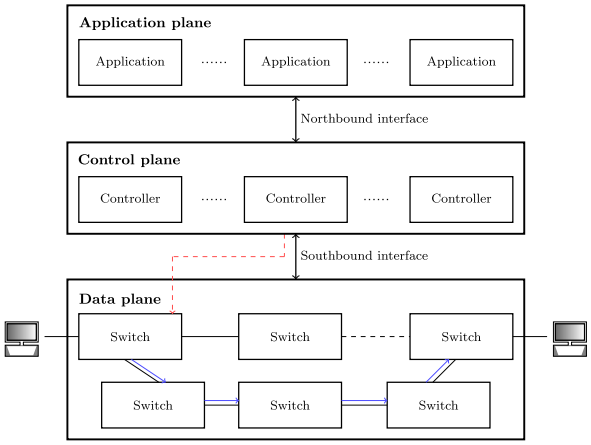


**FIGURE 15.** An example of proposed solution for adaptive routing: A switch receives information from the controller that there is a link failure in the primary path. Therefore, it redirects the traffic to an alternative path, which the controller recalculated.

Considering these approaches, a potential security solution should be able to, besides user and attacker dynamics, adapt to network dynamics too and in that way provide a high level of security under various network conditions.

### D. SUMMARY OF PROPOSED RESEARCH DIRECTIONS

Each of the adaptive mechanisms presented in previous sections is intended to primarily address one dynamic: adaptive encryption for user dynamics, adaptive function placement for attacker dynamics, and adaptive security-aware routing for network dynamics. However, they cannot be strictly separated, and should be combined in some situations to secure data plane more efficiently. Visual summary of presented research directions is shown in Fig. 16.

Some possible use cases in which combinations of the proposed adaptive mechanisms would be desirable are:

- Key management entity which generates keys for a particular user cannot provide him the desired key, i.e. cannot provide him the desired level of security at the moment. Therefore, if in the network exists another key management entity which can provide required keys, the user's request should be redirected to it. Although there are no network failures, and links towards the primary key management entity are stable, traffic should be redirected to another path, to meet user requirements. In this way, adaptive routing responds to user dynamics.
- User can sometimes require some specific device to check his traffic (e.g. IDS or firewall [197]). In this case, traffic should be redirected to a path that already contains a device with that function. If such path does not exist, i.e. a device or switch that performs that function does not exist in the network at the moment, the desired function should be placed on a certain switch

on a primary path. In this way, adaptive routing and adaptive function placement responds to user dynamics.

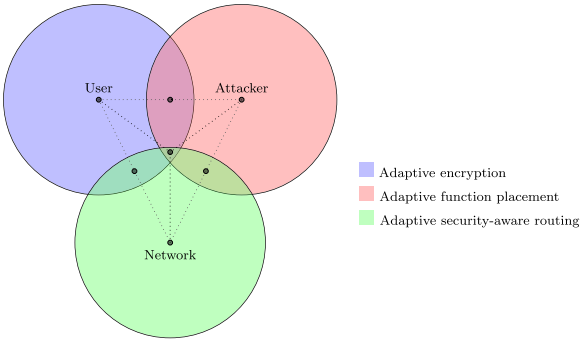


**FIGURE 16.** Summary of proposed research directions: three main solutions as responses to different dynamics in order to achieve adaptive security of data plane - adaptive encryption for dynamics of users, adaptive function placement for dynamics of attackers and adaptive security-aware routing for network dynamics. Each of the proposed solutions is primarily intended for one dynamic as it is mentioned, but should be combined with others to achieve a higher level of security.

The aforementioned use cases are just examples of possible situations which show that proposed adaptive mechanisms cannot be strictly separated, but should be combined to achieve a more efficient solution. Therefore, combination of adaptive encryption, adaptive function placement and adaptive security-aware routing is a promising approach towards a comprehensive and strong security solution for data plane.

However, deployment of such a comprehensive solution in the real-world will be challenging. Therefore, it should be considered how and where the proposed adaptive mechanisms can be deployed, with the aim of achieving both a high level of security and performance. Based on analyses presented in previous sections, the following can be stated:

- Communication with the controller is necessary in order to achieve efficient adaptive routing using a reactive approach. However, average recovery latency can be significantly increased due to flow insertions performed by the controller, especially in large-scale networks. Distributed control plane can help in reducing this latency and packet loss [192]. Additionally, global view of the controller can be combined with a programmable data plane and concept of segment routing. In that way, failure recovery can be achieved with acceptable latency, high scalability, low switch storage overhead and power consumption [195].
- Several proposed solutions for detection and mitigation of DoS attacks, which are based on ML and implemented on the controller, have shown potential for real-world deployments due to high detection accuracy, fast response and low resource consumption [96], [102], [106]. After detection, mitigation is performed by changing the flow rules in order to properly treat malicious traffic. Therefore, a certain switch will start to perform a specific function, i.e. to behave as some kind of simple firewall. Since the proposed solutions have shown acceptable performance, it should be considered to implement adaptive function placement through an application on the controller instead of deep network programmability due to lower complexity.

**TABLE 4.** List of abbreviations and acronyms.

| | |
|---|---|
| AAA | Authentication, Authorization, Accounting |
| ACK | Acknowledgement |
| AES | Advanced Encryption Standard |
| AI | Artificial Intelligence |
| ARP | Address Resolution Protocol |
| AT-RRM | Anomaly Triggered Random Routing Mutation |
| BGRU | Bidirectional Gated Recurrent Unit |
| BMv2 | Behavioral Model version 2 |
| BPNN | Back-Propagation Neural Network |
| CIA | Confidentiality, Integrity, Availability |
| CNN | Convolutional Neural Network |
| CPU | Central Processing Unit |
| DDPG | Deep Deterministic Policy Gradient |
| DH | Diffie-Hellman |
| DHCP | Dynamic Host Configuration Protocol |
| DL | Deep Learning |
| DNN | Deep Neural Network |
| DNS | Domain Name System |
| DPI | Deep Packet Inspection |
| DREAD | Damage, Reproducibility, Exploitability, Affected Users, Discoverability |
| DRL | Deep Reinforcement Learning |
| (D)DoS | (Distributed) Denial of Service |
| DT | Decision Tree |
| ECDH | Elliptic Curve Diffie-Hellman |
| EDR | Euclidean Distance Ratio |
| EFR | External Flow Rules |
| ETSI | European Telecommunications Standards Institute |
| FC | Flash Crowds |
| FL | Federated Learning |
| FPGA | Field Programmable Gate Array |
| GNN | Graph Neural Networks |
| IBC | Identity-Based Cryptography |
| ICMP | Internet Control Message Protocol |
| IDS | Intrusion Detection System |
| INT | In-band Network Telemetry |
| IoE/IoT | Internet of Everything/Internet of Things |
| IP | Internet Protocol |
| IPS | Intrusion Prevention System |
| ITS | Information Theoretical Security |
| ITU | International Telecommunication Union |
| KNN | K-Nearest Neighbors |
| KSI | Keyless Signature Infrastructure |
| LINDDUN | Linking, Identifying, Non-repudiation, Detecting, Data disclosure, Unawareness, Non-compliance |
| LLDP | Link Layer Discovery Protocol |
| LOFT | Low-Rate Flow Table Overflow |
| LR | Logistic Regression |
| LRCN | Long-term Recurrental Convolutional Network |
| LSTM | Long Short-Term Memory |
| LtR | Learning-to-Rank |
| MA-HS | Mayflies-Harmony Search |
| MAC | Media Access Control |
| MC-SDN | Multi-Controller SDN |
| MitM | Man-in-the-Middle |
| ML | Machine Learning |
| MLP | Multilayer Perceptron |
| MTD | Moving Target Defense |
| NB | Naive Bayes |
| NDPSO | Nonlinear Dynamic Particle Swarm Optimization |
| NFV | Network Function Virtualization |
| NPU | Neural Processing Unit |
| OCTAVE | Operationally Critical Threat, Asset, Vulnerability Evaluation |
| ODL | OpenDaylight |
| ONOS | Open Network Operating System |
| OTP | One-Time Pad |
| P2P | Peer-to-Peer |

**TABLE 4.** *(Continued.)* **List of abbreviations and acronyms.**

| | |
|---|---|
| PASTA | Process for Attack Simulation and Threat Analysis |
| PKG | Private Key Generator |
| POA | Preemptive Overflow Attack |
| POF | Protocol Oblivious Forwarding |
| PQC | Post-Quantum Cryptography |
| PSO | Particle Swarm Optimization |
| QKD | Quantum Key Distribution |
| QoS | Quality of Service |
| RF | Random Forest |
| RNN | Recurrent Neural Network |
| RRM | Random Route Mutation |
| SAE | Stacked autoencoder |
| SCADA | Supervisory Control and Data Acquisition |
| SDN | Software-Defined Network |
| SIEM | Security Information and Event Management |
| SMO | Sequential Minimal Optimization |
| SOM | Self-organizing Map |
| SSO-RM | Spatio-temporal Stochastic Optimization Route Mutation |
| STRIDE | Spoofing, Tampering, Repudiation, Information disclosure, Denial of service, Elevation of Privilege |
| SVM | Support Vector Machine |
| SVC | Support Vector Classifier |
| SYN | Synchronize |
| TCAM | Ternary Content-Addressable Memory |
| TCP | Transmission Control Protocol |
| TLS | Transport Layer Security |
| UDP | User Datagram Protocol |
| URL | Uniform Resource Locator |
| XOR | exclusive OR |
| XSS | Cross-Site Scripting |

- Hardware-based programmable encryptors in proposed solutions have shown high goodput, low latency and low power consumption [58], [59], [60]. These encryptors can adapt to different security requirements (i.e. achieve different levels of security) due to their programmability. Therefore, they should be considered as potential solutions to ensure adaptive encryption in real-world deployments. Determining the desired level of security has not gained much attention in the literature so far. Thus, in order to analyse possible ways to perform it in real-world deployments, further research needs to be done.

## VI. CONCLUSION

Next-generation networks will provide various services where security is especially important. Lack of security in new services such as telemedicine or autonomous driving can cause critical consequences. However, security is not a built-in network feature and thus requires special attention.

Data plane, as one of three main layers in software-defined networks, is not sufficiently analysed in existing literature, although it is a source of significant vulnerabilities which can compromise the entire network. Considering this, in our paper we have given a comprehensive survey on data plane security with possible directions for further security enhancement. Additionally, considering the variety of services requirements and possible complexity of the security situation in the network, we emphasise the significance of adaptivity in terms of security.

Compared to previous surveys, our survey gives a more detailed overview of possible attacks, threats, vulnerabilities and compromised security attributes in SDN data planes, and connections between them. It also gives a more exhaustive overview of the proposed security solutions, including security methods and implementation techniques that they used, with precise classification according to the security attributes they protect. This analysis helps to find possible mechanisms which will provide adaptivity in terms of security, which is particularly important in security of modern networks, but is not sufficiently analysed yet.

The presented survey aims to emphasise the importance of data plane security, providing a comprehensive, detailed basis for future research. Additionally, the presented future directions which utilise mechanisms that are already proposed in the literature can help in development of a new, strong and adaptive solution for further enhancement of data plane security.

## REFERENCES

[1] O. Flauzac, C. González, A. Hachani, and F. Nolot, "SDN based architecture for IoT and improvement of the security," in *Proc. IEEE 29th Int. Conf. Adv. Inf. Netw. Appl. Workshops*, Mar. 2015, pp. 688–693.
[2] L. Cui, F. R. Yu, and Q. Yan, "When big data meets software-defined networking: SDN for big data and big data for SDN," *IEEE Netw.*, vol. 30, no. 1, pp. 58–65, Jan. 2016.
[3] W. Xia, Y. Wen, C. H. Foh, D. Niyato, and H. Xie, "A survey on software-defined networking," *IEEE Commun. Surveys Tuts.*, vol. 17, no. 1, pp. 27–51, 1st Quart., 2015.
[4] D. Kreutz, F. M. V. Ramos, P. E. Veríssimo, C. E. Rothenberg, S. Azodolmolky, and S. Uhlig, "Software-defined networking: A comprehensive survey," *Proc. IEEE*, vol. 103, no. 1, pp. 14–76, Jan. 2015.
[5] N. Feamster, J. Rexford, and E. Zegura, "The road to SDN: An intellectual history of programmable networks," *ACM SIGCOMM Comput. Commun. Rev.*, vol. 44, no. 2, pp. 87–98, Apr. 2014.
[6] S. Sezer, S. Scott-Hayward, P. K. Chouhan, B. Fraser, D. Lake, J. Finnegan, N. Viljoen, M. Miller, and N. Rao, "Are we ready for SDN? Implementation challenges for software-defined networks," *IEEE Commun. Mag.*, vol. 51, no. 7, pp. 36–43, Jul. 2013.
[7] K. Benzekki, A. E. Fergougui, and A. E. B. E. Alaoui, "Software-defined networking (SDN): A survey," *Secur. Commun. Netw.*, vol. 9, no. 18, pp. 5803–5833, Dec. 2016.
[8] S. A. Abdel Hakeem, H. H. Hussein, and H. Kim, "Security requirements and challenges of 6G technologies and applications," *Sensors*, vol. 22, no. 5, p. 1969, Mar. 2022.
[9] V.-L. Nguyen, P.-C. Lin, B.-C. Cheng, R.-H. Hwang, and Y.-D. Lin, "Security and privacy for 6G: A survey on prospective technologies and challenges," *IEEE Commun. Surveys Tuts.*, vol. 23, no. 4, pp. 2384–2428, 2021.
[10] W. Braun and M. Menth, "Software-defined networking using OpenFlow: Protocols, applications and architectural design choices," *Future Internet*, vol. 6, no. 2, pp. 302–336, May 2014.
[11] H. H. H. Mahmoud, A. A. Amer, and T. Ismail, "6G: A comprehensive survey on technologies, applications, challenges, and research problems," *Trans. Emerg. Telecommun. Technol.*, vol. 32, no. 4, p. 4233, Apr. 2021.
[12] D. Kreutz, F. M. V. Ramos, and P. Veríssimo, "Towards secure and dependable software-defined networks," in *Proc. 2nd ACM SIGCOMM workshop Hot topics Softw. defined Netw.* New York, NY, USA: Association for Computing Machinery, Aug. 2013, p. 55.
[13] M. Antikainen, T. Aura, and M. Särelä, "Spook in your network: Attacking an SDN with a compromised OpenFlow switch," in *Proc. Secure IT Syst., 19th Nordic Conf.*, Norway, Jan. 2014, pp. 229–244.
[14] I. Ahmad, S. Namal, M. Ylianttila, and A. Gurtov, "Security in software defined networks: A survey," *IEEE Commun. Surveys Tuts.*, vol. 17, no. 4, pp. 2317–2346, 4th Quart., 2015.
[15] I. Alsmadi and D. Xu, "Security of software defined networks: A survey," *Comput. Secur.*, vol. 53, pp. 79–108, Sep. 2015.

[16] J. C. Correa Chica, J. C. Imbachi, and J. F. Botero Vega, "Security in SDN: A comprehensive survey," *J. Netw. Comput. Appl.*, vol. 159, Jun. 2020, Art. no. 102595.

[17] S. Scott-Hayward, S. Natarajan, and S. Sezer, "A survey of security in software defined networks," *IEEE Commun. Surveys Tuts.*, vol. 18, no. 1, pp. 623–654, 1st Quart., 2016.

[18] M. Rahouti, K. Xiong, Y. Xin, S. K. Jagatheesaperumal, M. Ayyash, and M. Shaheed, "SDN security review: Threat taxonomy, implications, and open challenges," *IEEE Access*, vol. 10, pp. 45820–45854, 2022.

[19] M. S. Farooq, S. Riaz, and A. Alvi, "Security and privacy issues in software-defined networking (SDN): A systematic literature review," *Electronics*, vol. 12, no. 14, p. 3077, Jul. 2023.

[20] R. Deb and S. Roy, "A comprehensive survey of vulnerability and information security in SDN," *Comput. Netw.*, vol. 206, Apr. 2022, Art. no. 108802.

[21] M. B. Jiménez, D. Fernández, J. E. Rivadeneira, L. Bellido, and A. Cárdenas, "A survey of the main security issues and solutions for the SDN architecture," *IEEE Access*, vol. 9, pp. 122016–122038, 2021.

[22] N. Dayal, P. Maity, S. Srivastava, and R. Khondoker, "Research trends in security and DDoS in SDN," *Secur. Commun. Netw.*, vol. 9, no. 18, pp. 6386–6411, Dec. 2016.

[23] A. H. Abdi, L. Audah, A. Salh, M. A. Alhartomi, H. Rasheed, S. Ahmed, and A. Tahir, "Security control and data planes of SDN: A comprehensive review of traditional, AI, and MTD approaches to security solutions," *IEEE Access*, vol. 12, pp. 69941–69980, 2024.

[24] M. Shahzad, S. Rizvi, T. A. Khan, S. Ahmad, and A. A. Ateya, "An exhaustive parametric analysis for securing SDN through traditional, AI/ML, and blockchain approaches: A systematic review," *Int. J. Netw. Distrib. Comput.*, vol. 13, no. 1, pp. 1–16, Jun. 2025.

[25] A. Shaghaghi, M. A. Kaafar, R. Buyya, and S. Jha, "Software-defined network (SDN) data plane security: Issues, solutions, and future directions," in *Handbook of Computer Networks and Cyber Security*, 2020, pp. 341–387.

[26] T. Dargahi, A. Caponi, M. Ambrosin, G. Bianchi, and M. Conti, "A survey on the security of stateful SDN data planes," *IEEE Commun. Surveys Tuts.*, vol. 19, no. 3, pp. 1701–1725, 3rd Quart., 2017.

[27] A. N. Alhaj and N. Dutta, "Analysis of security attacks in SDN network: A comprehensive survey," in *Proc. Contemp. Issues Commun.*, Nov. 2021, pp. 27–37.

[28] A. Pradhan and R. Mathew, "Solutions to vulnerabilities and threats in software defined networking (SDN)," in *Proc. Comput. Sci.*, vol. 171, Jan. 2020, pp. 2581–2589.

[29] P. Krishnan and J. S. Najeem, "A review of security, threats and mitigation approaches for SDN architecture," *Int. J. Innov. Technol. Exploring Eng.*, vol. 8, no. 5, pp. 389–393, May 2019.

[30] M. Iqbal, F. Iqbal, F. Mohsin, M. Rizwan, and F. Ahmad, "Security issues in software defined networking (SDN): Risks, challenges and potential solutions," *Int. J. Adv. Comput. Sci. Appl.*, vol. 10, no. 10, pp. 298–303, Oct. 2019.

[31] M. Dabbagh, B. Hamdaoui, M. Guizani, and A. Rayes, "Software-defined networking security: Pros and cons," *IEEE Commun. Mag.*, vol. 53, no. 6, pp. 73–79, Jun. 2015.

[32] I. A. Mahar, W. Libing, Z. A. Maher, and G. A. Rahu, "A comprehensive survey of software defined networking and its security threats," in *Proc. IEEE 1st Karachi Sect. Humanitarian Technol. Conf. (KHI-HTC)*, Jan. 2024, pp. 1–5.

[33] *Microsoft Threat Modeling Tool Threats*. Accessed: Nov. 16, 2023. [Online]. Available: https://learn.microsoft.com/en-us/azure/security/develop/threat-modeling-tool-threats

[34] L. O. Nweke, "Using the CIA and AAA models to explain cybersecurity activities," *PM World J.*, pp. 1–3, Jan. 2017.

[35] M. Deng, K. Wuyts, R. Scandariato, B. Preneel, and W. Joosen, "A privacy threat analysis framework: Supporting the elicitation and fulfillment of privacy requirements," *Requirements Eng.*, vol. 16, no. 1, pp. 3–32, Mar. 2011.

[36] M. Howard and D. LeBlanc, *Writing Secure Code*. London, U.K.: Pearson Education, 2003.

[37] T. UcedaVelez and M. M. Morana, *Risk Centric Threat Modeling: Process for Attack Simulation and Threat Analysis*. Hoboken, NJ, USA: Wiley, 2015.

[38] C. J. Alberts, S. G. Behrens, R. D. Pethia, and W. R. Wilson, "Operationally critical threat, asset, and vulnerability evaluation (OCTAVE) framework, version 1.0," Carnegie Mellon Univ., Pittsburgh, PA, USA, Tech. Rep. CMU/SEI-99-TR-017, Sep. 1999.

[39] N. Naik, P. Jenkins, P. Grace, D. Naik, S. Prajapat, and J. Song, "A comparative analysis of threat modelling methods: STRIDE, DREAD, VAST, PASTA, OCTAVE, and LINDDUN," in *Proc. Int. Conf. Comput.*, Oct. 2024, pp. 271–280.

[40] K. S. Suhas, "Evaluation of threat models," *Int. J. Res. Appl. Sci. Eng. Technol.*, vol. 11, no. 3, pp. 809–813, Mar. 2023.

[41] A. Aseeri, N. Netjinda, and R. Hewett, "Alleviating eavesdropping attacks in software-defined networking data plane," in *Proc. 12th Annu. Conf. Cyber Inf. Secur. Res.*, Apr. 2017, pp. 1–8.

[42] S. Gao, Z. Li, B. Xiao, and G. Wei, "Security threats in the data plane of software-defined networks," *IEEE Netw.*, vol. 32, no. 4, pp. 108–113, Jul. 2018.

[43] M. F. Hyder and M. A. Ismail, "Securing control and data planes from reconnaissance attacks using distributed shadow controllers, reactive and proactive approaches," *IEEE Access*, vol. 9, pp. 21881–21894, 2021.

[44] A. Shaghaghi, M. A. Kaafar, and S. Jha, "WedgeTail: An intrusion prevention system for the data plane of software defined networks," in *Proc. ACM Asia Conf. Comput. Commun. Secur.*, Apr. 2017, pp. 849–861.

[45] X. Wu, M. Liu, W. Dou, and S. Yu, "DDoS attacks on data plane of software-defined network: Are they possible?" *Secur. Commun. Netw.*, vol. 9, no. 18, pp. 5444–5459, Dec. 2016.

[46] E. G. da Silva, L. A. Dias Knob, J. A. Wickboldt, L. P. Gaspary, L. Z. Granville, and A. Schaeffer-Filho, "Capitalizing on SDN-based SCADA systems: An anti-eavesdropping case-study," in *Proc. IFIP/IEEE Int. Symp. Integr. Netw. Manage. (IM)*, May 2015, pp. 165–173.

[47] X. Xu, H. Hu, Y. Liu, J. Tan, H. Zhang, and H. Song, "Moving target defense of routing randomization with deep reinforcement learning against eavesdropping attack," *Digit. Commun. Netw.*, vol. 8, no. 3, pp. 373–387, Jun. 2022.

[48] Q. Duan, E. Al-Shaer, and H. Jafarian, "Efficient random route mutation considering flow and network constraints," in *Proc. IEEE Conf. Commun. Netw. Secur. (CNS)*, Oct. 2013, pp. 260–268.

[49] J. Liu, H. Zhang, and Z. Guo, "A defense mechanism of random routing mutation in SDN," *IEICE Trans. Inf. Syst.*, vol. 100, no. 5, pp. 1046–1054, 2017.

[50] Z. Zhou, C. Xu, X. Kuang, T. Zhang, and L. Sun, "An efficient and agile spatio-temporal route mutation moving target defense mechanism," in *Proc. IEEE Int. Conf. Commun. (ICC)*, May 2019, pp. 1–6.

[51] F. Jiang, "Combat-sniff: A comprehensive countermeasure to resist data plane eavesdropping in software-defined networks," *Amer. J. Netw. Commun.*, vol. 5, no. 2, pp. 27–34, 2016.

[52] G. Liu, W. Quan, N. Cheng, N. Lu, H. Zhang, and X. Shen, "P4NIS: Improving network immunity against eavesdropping with programmable data planes," in *Proc. IEEE Conf. Comput. Commun. Workshops (INFOCOM WKSHPS)*, Jul. 2020, pp. 91–96.

[53] G. Liu, W. Quan, N. Cheng, D. Gao, N. Lu, H. Zhang, and X. Shen, "Softwarized IoT network immunity against eavesdropping with programmable data planes," *IEEE Internet Things J.*, vol. 8, no. 8, pp. 6578–6590, Apr. 2021.

[54] I. Oliveira, E. Neto, R. Immich, R. Fontes, A. Neto, F. Rodriguez, and C. E. Rothenberg, "Dh-aes-p4: On-premise encryption and in-band key-exchange in P4 fully programmable data planes," in *Proc. IEEE Conf. Netw. Function Virtualization Softw. Defined Netw. (NFV-SDN)*, Nov. 2021, pp. 148–153.

[55] T. Adhikari, M. Kule, and A. K. Khan, "An ECDH and AES based encryption approach for prevention of MiTM in SDN southbound communication interface," in *Proc. 13th Int. Conf. Comput. Commun. Netw. Technol. (ICCCNT)*, Oct. 2022, pp. 1–5.

[56] J. H. Lam, S. Lee, H.-J. Lee, and Y. E. Oktian, "Securing SDN southbound and data plane communication with IBC," *Mobile Inf. Syst.*, vol. 2016, pp. 1–12, Jan. 2016.

[57] C. Peng, Q. Zhang, and C. Tang, "Improved TLS handshake protocols using identity-based cryptography," in *Proc. Int. Symp. Inf. Eng. Electron. Commerce*, May 2009, pp. 135–139.

[58] E. Arabul, R. S. Tessinari, O. Alia, E. Hugues-Salas, G. T. Kanellos, R. Nejabati, and D. Simeonidou, "Experimental demonstration of programmable 100 Gb/s SDN-enabled encryptors/decryptors for QKD networks," in *Proc. Opt. Fiber Commun. Conf. Exhib. (OFC)*, Jun. 2021, pp. 1–3.

[59] E. Arabul, R. S. Tessinari, O. Alia, R. Oliveira, G. T. Kanellos, R. Nejabati, and D. Simeonidou, "100 Gb/s dynamically programmable SDN-enabled hardware encryptor for optical networks," *J. Opt. Commun. Netw.*, vol. 14, no. 1, pp. A50–A60, Jan. 2022.

[60] R. S. Tessinari, E. Arabul, O. Alia, A. S. Muqaddas, G. T. Kanellos, R. Nejabati, and D. Simeonidou, "Demonstration of a dynamic QKD network control using a QKD-aware SDN application over a programmable hardware encryptor," in *Proc. Opt. Fiber Commun. Conf. Exhib. (OFC)*, Jun. 2021, pp. 1–3.

[61] Y. Peng, C. Wu, B. Zhao, W. Yu, B. Liu, and S. Qiao, "QKDFlow: QKD based secure communication towards the OpenFlow interface in SDN," in *Proc. 4th Annu. Int. Conf. Geo-Informatics Resource Manage. Sustain. Ecosystem (GRMSE)*, Jan. 2017, pp. 410–415.

[62] S. S. Mahdi and A. A. Abdullah, "Enhanced security of software-defined network and network slice through hybrid quantum key distribution protocol," *Infocommun. J.*, vol. 14, no. 3, pp. 9–15, 2022.

[63] C. R. García, S. Rommel, J. J. V. Olmos, and I. T. Monroy, "Enhancing the security of software defined networks via quantum key distribution and post-quantum cryptography," in *Proc. Int. Symp. Distrib. Comput. Artif. Intell.*, Jan. 2023, pp. 428–437.

[64] M. H. Rempola, A. Smith, Y. Li, and L. Du, "Securing SDN communication through quantum key distribution," in *Proc. IEEE Transp. Electrific. Conf. Expo (ITEC)*, Jun. 2024, pp. 1–5.

[65] 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, pp. 58845–58858, 2020.

[66] F. Hauser, M. Häberle, M. Schmidt, and M. Menth, "P4-IPsec: Site-to-site and host-to-site VPN with IPsec in P4-based SDN," *IEEE Access*, vol. 8, pp. 139567–139586, 2020.

[67] M. Mehic, L. Michalek, E. Dervisevic, P. Burdiak, M. Plakalovic, J. Rozhon, N. Mahovac, F. Richter, E. Kaljic, F. Lauterbach, P. Njemcevic, A. Maric, M. Hamza, P. Fazio, and M. Voznak, "Quantum cryptography in 5G networks: A comprehensive overview," *IEEE Commun. Surveys Tuts.*, vol. 26, no. 1, pp. 302–346, 1st Quart., 2024.

[68] A. AlSabeh, E. Kfoury, J. Crichigno, and E. Bou-Harb, "P4DDPI: Securing P4-programmable data plane networks via DNS deep packet inspection," in *Proc. Workshop Meas., Attacks, Defenses Web*, 2022, pp. 1–7.

[69] *Pfsense*. Accessed: Mar. 10, 2025. [Online]. Available: https://www.pfsense.org/

[70] N. Narayanan, G. C. Sankaran, and K. M. Sivalingam, "Mitigation of security attacks in the SDN data plane using P4-enabled switches," in *Proc. IEEE Int. Conf. Adv. Netw. Telecommun. Syst. (ANTS)*, Dec. 2019, pp. 1–6.

[71] S. Buzura, M. Lehene, B. Iancu, and V. Dadarlat, "An extendable software architecture for mitigating ARP spoofing-based attacks in SDN data plane layer," *Electronics*, vol. 11, no. 13, p. 1965, Jun. 2022.

[72] X. Guo, C. Wang, L. Cao, Y. Jiang, and Y. Yan, "A novel security mechanism for software defined network based on blockchain," *Comput. Sci. Inf. Syst.*, vol. 19, no. 2, pp. 523–545, 2022.

[73] A. Almaini, A. Al-Dubai, I. Romdhani, and M. Schramm, "Delegation of authentication to the data plane in software-defined networks," in *Proc. IEEE Int. Conferences Ubiquitous Comput. Commun. (IUCC) Data Sci. Comput. Intell. (DSCI) Smart Comput., Netw. Services (SmartCNS)*, Oct. 2019, pp. 58–65.

[74] J. Yao, Z. Han, M. Sohail, and L. Wang, "A robust security architecture for SDN-based 5G networks," *Future Internet*, vol. 11, no. 4, p. 85, Mar. 2019.

[75] J. Kempf, E. Bellagamba, A. Kern, D. Jocha, A. Takacs, and P. Sköldström, "Scalable fault management for OpenFlow," in *Proc. IEEE Int. Conf. Commun. (ICC)*, Jun. 2012, pp. 6606–6610.

[76] A. Darabseh, M. Al-Ayyoub, Y. Jararweh, E. Benkhelifa, M. Vouk, and A. Rindos, "SDSecurity: A software defined security experimental framework," in *Proc. IEEE Int. Conf. Commun. Workshop (ICCW)*, Jun. 2015, pp. 1871–1876.

[77] D. Scholz, A. Oeldemann, F. Geyer, S. Gallenmüller, H. Stubbe, T. Wild, A. Herkersdorf, and G. Carle, "Cryptographic hashing in P4 data planes," in *Proc. ACM/IEEE Symp. Archit. Netw. Commun. Syst. (ANCS)*, Sep. 2019, pp. 1–6.

[78] G. Hessam, G. Saba, and M. I. Alkhayat, "A new approach for detecting violation of data plane integrity in software defined networks," *J. Comput. Secur.*, vol. 29, no. 3, pp. 341–358, May 2021.

[79] T.-W. Chao, Y.-M. Ke, B.-H. Chen, J.-L. Chen, C. J. Hsieh, S.-C. Lee, and H.-C. Hsiao, "Securing data planes in software-defined networks," in *Proc. IEEE NetSoft Conf. Workshops (NetSoft)*, Jun. 2016, pp. 465–470.

[80] D. Smyth, S. Scott-Hayward, V. Cionca, S. McSweeney, and D. O'Shea, "SECAP switch–defeating topology poisoning attacks using P4 data planes," *J. Netw. Syst. Manage.*, vol. 31, no. 1, p. 28, Jan. 2023.

[81] M. Dhawan, R. Poddar, K. S. Mahajan, and V. Mann, "SPHINX: Detecting security attacks in software-defined networks," in *Proc. Netw. Distrib. Syst. Secur. Symp.*, 2015, pp. 8–11.

[82] S. Mahrach and A. Haqiq, "DDoS flooding attack mitigation in software defined networks," *Int. J. Adv. Comput. Sci. Appl.*, vol. 11, no. 1, pp. 693–700, 2020.

[83] Y. Afek, A. Bremler-Barr, and L. Shafir, "Network anti-spoofing with SDN data plane," in *Proc. IEEE Conf. Comput. Commun. (INFOCOM)*, May 2017, pp. 1–9.

[84] D. Scholz, S. Gallenmüller, H. Stubbe, and G. Carle, "SYN flood defense in programmable data planes," in *Proc. 3rd P4 Workshop Eur.*, Dec. 2020, pp. 13–20.

[85] P.-W. Chi, C.-T. Kuo, J.-W. Guo, and C.-L. Lei, "How to detect a compromised SDN switch," in *Proc. 1st IEEE Conf. Netw. Softwarization (NetSoft)*, Apr. 2015, pp. 1–6.

[86] R. Datta, S. Choi, A. Chowdhary, and Y. Park, "P4Guard: Designing P4 based firewall," in *Proc. MILCOM - IEEE Mil. Commun. Conf. (MILCOM)*, Oct. 2018, pp. 1–6.

[87] J. Deng, H. Hu, H. Li, Z. Pan, K.-C. Wang, G.-J. Ahn, J. Bi, and Y. Park, "VNGuard: An NFV/SDN combination framework for provisioning and managing virtual firewalls," in *Proc. IEEE Conf. Netw. Function Virtualization Softw. Defined Netw. (NFV-SDN)*, Nov. 2015, pp. 107–114.

[88] P. Vörös and A. Kiss, "Security middleware programming using P4," in *Proc. Int. Conf. Hum. Aspects Inf. Security, Privacy, Trust*, Jan. 2016, pp. 277–287.

[89] B. A. Reddy, K. S. Sahoo, and M. Bhuyan, "Securing P4-SDN data plane against flow table modification attack," in *Proc. NOMS - IEEE Netw. Oper. Manage. Symp.*, May 2024, pp. 1–5.

[90] F. Musumeci, V. Ionata, F. Paolucci, F. Cugini, and M. Tornatore, "Machine-learning-assisted DDoS attack detection with P4 language," in *Proc. IEEE Int. Conf. Commun. (ICC)*, Jun. 2020, pp. 1–6.

[91] W. Zhang, S. Jing, and C. Zhao, "Anti-DDoS attacks strategy of SDN data plane with data augmentation based on P4," in *Proc. IEEE Int. Conf. High Perform. Comput., Commun. Data Sci. Syst., Smart City Dependability Sensor, Cloud Big Data Syst. Appl. (HPCC/DSS/SmartCity/DependSys)*, Dec. 2023, pp. 348–354.

[92] R. Doriguzzi-Corin, L. A. D. Knob, L. Mendozzi, D. Siracusa, and M. Savi, "Introducing packet-level analysis in programmable data planes to advance network intrusion detection," *Comput. Netw.*, vol. 239, Feb. 2024, Art. no. 110162.

[93] A. Ganesan and K. Sarac, "Attack detection and mitigation using intelligent data planes in SDNs," in *Proc. IEEE Global Commun. Conf. (GLOBECOM)*, Dec. 2022, pp. 1–6.

[94] T. Abhiroop, S. Babu, and B. S. Manoj, "A machine learning approach for detecting DoS attacks in SDN switches," in *Proc. Twenty 4th Nat. Conf. Commun. (NCC)*, Feb. 2018, pp. 1–6.

[95] B. Celesova, J. Val'ko, R. Grezo, and P. Helebrandt, "Enhancing security of SDN focusing on control plane and data plane," in *Proc. 7th Int. Symp. Digit. Forensics Secur. (ISDFS)*, Jun. 2019, pp. 1–6.

[96] W. G. Gadallah, H. M. Ibrahim, and N. M. Omar, "A deep learning technique to detect distributed denial of service attacks in software-defined networks," *Comput. Secur.*, vol. 137, Feb. 2024, Art. no. 103588.

[97] S. Gao, Z. Peng, B. Xiao, A. Hu, Y. Song, and K. Ren, "Detection and mitigation of DoS attacks in software defined networks," *IEEE/ACM Trans. Netw.*, vol. 28, no. 3, pp. 1419–1433, Jun. 2020.

[98] M. Yue, H. Wang, L. Liu, and Z. Wu, "Detecting DoS attacks based on multi-features in SDN," *IEEE Access*, vol. 8, pp. 104688–104700, 2020.

[99] M. Sheng, H. Liu, X. Yang, W. Wang, J. Huang, and B. Wang, "Network security situation prediction in software defined networking data plane," in *Proc. IEEE Int. Conf. Adv. Electr. Eng. Comput. Appl. (AEECA)*, Aug. 2020, pp. 475–479.

[100] J. Cao, M. Xu, Q. Li, K. Sun, Y. Yang, and J. Zheng, "Disrupting SDN via the data plane: A low-rate flow table overflow attack," in *Proc. Int. Conf. Secur. Privacy Commun. Syst.*, Apr. 2018, pp. 356–376.

[101] J. Cao, M. Xu, Q. Li, K. Sun, and Y. Yang, "The LOFT attack: Overflowing SDN flow tables at a low rate," *IEEE/ACM Trans. Netw.*, vol. 31, no. 3, pp. 1416–1431, Jun. 2023.

[102] D. Tang, Y. Yan, C. Gao, W. Liang, and W. Jin, "LtRFT: Mitigate the low-rate data plane DDoS attack with learning-to-rank enabled flow tables," *IEEE Trans. Inf. Forensics Security*, vol. 18, pp. 3143–3157, 2023.

[103] D. Tang, Z. Zheng, K. Li, C. Yin, W. Liang, and J. Zhang, "FTOP: An efficient flow table overflow preventing system for switches in SDN," *IEEE Trans. Netw. Sci. Eng.*, vol. 11, no. 3, pp. 2524–2536, May 2024.

[104] D. Tang, C. Gao, W. Liang, J. Zhang, and K. Li, "FTMaster: A detection and mitigation system of low-rate flow table overflow attacks via SDN," *IEEE Trans. Netw. Service Manage.*, vol. 20, no. 4, pp. 5073–5084, Dec. 2023.

[105] D. Tang, D. Zhang, Z. Qin, Q. Yang, and S. Xiao, "SFTO-guard: Real-time detection and mitigation system for slow-rate flow table overflow attacks," *J. Netw. Comput. Appl.*, vol. 213, Apr. 2023, Art. no. 103597.

[106] D. Tang, Z. Zheng, C. Yin, B. Xiong, Z. Qin, and Q. Yang, "FTODefender: An efficient flow table overflow attacks defending system in SDN," *Expert Syst. Appl.*, vol. 237, Mar. 2024, Art. no. 121460.

[107] A. Mudgal, A. Verma, M. Singh, K. S. Sahoo, E. Elmroth, and M. Bhuyan, "FloRa: Flow table low-rate overflow reconnaissance and detection in SDN," *IEEE Trans. Netw. Service Manage.*, vol. 21, no. 6, pp. 6670–6683, Dec. 2024.

[108] L. Jain, V. U., and S. Vollala, "FTSheild: An intelligent framework for LOFT attack detection and mitigation with programmable data plane," *Expert Syst. Appl.*, vol. 265, Mar. 2025, Art. no. 125865.

[109] Y. Zeng, Y. Wang, and Y. Liu, "Research on detection and mitigation methods of adaptive flow table overflow attacks in software-defined networks," *IEEE Access*, vol. 12, pp. 48830–48845, 2024.

[110] Y. Liu, Y. Wang, and H. Feng, "POAGuard: A defense mechanism against preemptive table overflow attack in software-defined networks," *IEEE Access*, vol. 11, pp. 123659–123676, 2023.

[111] R. Neres Carvalho, J. Luiz Bordim, and E. Adilio Pelinson Alchieri, "Entropy-based DoS attack identification in SDN," in *Proc. IEEE Int. Parallel Distrib. Process. Symp. Workshops (IPDPSW)*, May 2019, pp. 627–634.

[112] R. N. Carvalho, L. R. Costa, J. L. Bordim, and E. A. P. Alchieri, "Enhancing an SDN architecture with DoS attack detection mechanisms," *Adv. Sci., Technol. Eng. Syst. J.*, vol. 5, no. 2, pp. 215–224, 2020.

[113] D. Ding, M. Savi, and D. Siracusa, "Tracking normalized network traffic entropy to detect DDoS attacks in P4," *IEEE Trans. Dependable Secure Comput.*, vol. 19, no. 6, pp. 4019–4031, Nov. 2022.

[114] L. A. Q. González, 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 *Proc. IFIP/IEEE Int. Symp. Integr. Netw. Manage. (IM)*, May 2021, pp. 393–401.

[115] G. Gori, L. Rinieri, A. Al Sadi, A. Melis, F. Callegati, and M. Prandini, "GRAPH4: A security monitoring architecture based on data plane anomaly detection metrics calculated over attack graphs," *Future Internet*, vol. 15, no. 11, p. 368, Nov. 2023.

[116] L. Castanheira, R. Parizotto, and A. E. Schaeffer-Filho, "FlowStalker: Comprehensive traffic flow monitoring on the data plane using P4," in *Proc. IEEE Int. Conf. Commun. (ICC)*, May 2019, pp. 1–6.

[117] R. Durner, C. Lorenz, M. Wiedemann, and W. Kellerer, "Detecting and mitigating denial of service attacks against the data plane in software defined networks," in *Proc. IEEE Conf. Netw. Softwarization (NetSoft)*, Jul. 2017, pp. 1–6.

[118] R.-H. Hwang, V.-L. Nguyen, and P.-C. Lin, "StateFit: A security framework for SDN programmable data plane model," in *Proc. 15th Int. Symp. Pervasive Syst., Algorithms Netw. (I-SPAN)*, Oct. 2018, pp. 168–173.

[119] G. Shang, P. Zhe, X. Bin, H. Aiqun, and R. Kui, "FloodDefender: Protecting data and control plane resources under SDN-aimed DoS attacks," in *Proc. IEEE Conf. Comput. Commun. (INFOCOM)*, May 2017, pp. 1–9.

[120] E. Kaljic, A. Maric, and P. Njemcevic, "DoS attack mitigation in SDN networks using a deeply programmable packet-switching node based on a hybrid FPGA/CPU data plane architecture," in *Proc. XXVII Int. Conf. Inf., Commun. Autom. Technol. (ICAT)*, Oct. 2019, pp. 1–6.

[121] S. Wang, S. Chandrasekharan, K. Gomez, S. Kandeepan, A. Al-Hourani, M. R. Asghar, G. Russello, and P. Zanna, "SECOD: SDN sEcure control and data plane algorithm for detecting and defending against DoS attacks," in *Proc. NOMS-IEEE/IFIP Netw. Oper. Manage. Symp.*, Apr. 2018, pp. 1–5.

[122] A. Yazdinejad, R. M. Parizi, A. Dehghantanha, and K.-K.-R. Choo, "P4-to-blockchain: A secure blockchain-enabled packet parser for software defined networking," *Comput. Secur.*, vol. 88, Jan. 2020, Art. no. 101629.

[123] D. Das, S. Banerjee, K. Dasgupta, P. Chatterjee, U. Ghosh, and U. Biswas, "Blockchain enabled SDN framework for security management in 5G applications," in *Proc. 24th Int. Conf. Distrib. Comput. Netw.*, Jan. 2023, pp. 414–419.

[124] R. Taheri, H. Ahmed, and E. Arslan, "Deep learning for the security of software-defined networks: A review," *Cluster Comput.*, vol. 26, no. 5, pp. 3089–3112, Oct. 2023.

[125] T. Farasat, J. Kim, and J. Posegga, "Advancing network security: A comprehensive testbed and dataset for machine learning-based intrusion detection," 2024, *arXiv:2410.18332*.

[126] R. T. El-Maghraby, N. M. Abd Elazim, and A. M. Bahaa-Eldin, "A survey on deep packet inspection," in *Proc. 12th Int. Conf. Comput. Eng. Syst. (ICCES)*, Dec. 2017, pp. 188–197.

[127] C. Cho, J. Lee, E.-D. Kim, and J.-d. Ryoo, "A sophisticated packet forwarding scheme with deep packet inspection in an OpenFlow switch," in *Proc. Int. Conf. Softw. Netw. (ICSN)*, May 2016, pp. 1–5.

[128] D. Sanvito, D. Moro, and A. Capone, "Towards traffic classification offloading to stateful SDN data planes," in *Proc. IEEE Conf. Netw. Softwarization (NetSoft)*, Jul. 2017, pp. 1–4.

[129] H. Huang, P. Li, and S. Guo, "Traffic scheduling for deep packet inspection in software-defined networks," *Concurrency Comput., Pract. Exper.*, vol. 29, no. 16, p. 3967, Aug. 2017.

[130] Y.-D. Lin, P.-C. Lin, C.-H. Yeh, Y.-C. Wang, and Y.-C. Lai, "An extended SDN architecture for network function virtualization with a case study on intrusion prevention," *IEEE Netw.*, vol. 29, no. 3, pp. 48–53, May 2015.

[131] J. Hypolite, J. Sonchack, S. Hershkop, N. Dautenhahn, A. DeHon, and J. M. Smith, "DeepMatch: Practical deep packet inspection in the data plane using network processors," in *Proc. 16th Int. Conf. Emerg. Netw. EXperiments Technol.*, Nov. 2020, pp. 336–350.

[132] T. Park, Y. Kim, V. Yegneswaran, P. Porras, Z. Xu, K. Park, and S. Shin, "DPX: Data-plane eXtensions for SDN security service instantiation," in *Proc. Detection Intrusions Malware*, Gothenburg, Sweden, Jan. 2019, pp. 415–437.

[133] J. Kim, Y. Kim, V. Yegneswaran, P. Porras, S. Shin, and T. Park, "Extended data plane architecture for in-network security services in software-defined networks," *Comput. Secur.*, vol. 124, Jan. 2023, Art. no. 102976.

[134] S. Gupta, D. Gosain, M. Kwon, and H. B. Acharya, "DeeP4R: Deep packet inspection in P4 using packet recirculation," in *Proc. IEEE Conf. Comput. Commun. (INFOCOM)*, May 2023, pp. 1–10.

[135] M. Sainz, I. Garitano, M. Iturbe, and U. Zurutuza, "Deep packet inspection for intelligent intrusion detection in software-defined industrial networks: A proof of concept," *Log. J. IGPL*, vol. 28, no. 4, pp. 461–472, Jul. 2020.

[136] M. Bouet, J. Leguay, and V. Conan, "Cost-based placement of virtualized deep packet inspection functions in SDN," in *Proc. MILCOM-IEEE Mil. Commun. Conf.*, Nov. 2013, pp. 992–997.

[137] Q. Cheng, C. Wu, H. Zhou, D. Kong, D. Zhang, J. Xing, and W. Ruan, "Machine learning based malicious payload identification in software-defined networking," *J. Netw. Comput. Appl.*, vol. 192, Oct. 2021, Art. no. 103186.

[138] M. Dimolianis, A. Pavlidis, and V. Maglaris, "Signature-based traffic classification and mitigation for DDoS attacks using programmable network data planes," *IEEE Access*, vol. 9, pp. 113061–113076, 2021.

[139] C. Yu, J. Lan, J. Xie, and Y. Hu, "QoS-aware traffic classification architecture using machine learning and deep packet inspection in SDNs," *Proc. Comput. Sci.*, vol. 131, pp. 1209–1216, Jan. 2018.

[140] P. Krishnan, S. Duttagupta, and K. Achuthan, "VARMAN: Multi-plane security framework for software defined networks," *Comput. Commun.*, vol. 148, pp. 215–239, Dec. 2019.

[141] T. V. Phan, T. G. Nguyen, N.-N. Dao, T. T. Huong, N. H. Thanh, and T. Bauschert, "DeepGuard: Efficient anomaly detection in SDN with fine-grained traffic flow monitoring," *IEEE Trans. Netw. Service Manage.*, vol. 17, no. 3, pp. 1349–1362, Sep. 2020.

[142] A. Ahmad, E. Harjula, M. Ylianttila, and I. Ahmad, "Evaluation of machine learning techniques for security in SDN," in *Proc. IEEE Globecom Workshops (GC Wkshps)*, Dec. 2020, pp. 1–6.

[143] B. Brandino, P. Casas, and E. Grampín, "Detecting attacks at switching speed: Ai/Ml and active learning for in-network monitoring in data planes," in *Proc. IEEE 32nd Int. Conf. Netw. Protocols (ICNP)*, Oct. 2024, pp. 1–6.

[144] B. Brandino, E. Grampin, K. Dietz, N. Wehner, M. Seufert, T. Hoßfeld, and P. Casas, "HALIDS: A hardware-assisted machine learning IDS for in-network monitoring," in *Proc. 8th Netw. Traffic Meas. Anal. Conf. (TMA)*, May 2024, pp. 1–4.

[145] A. Alshamrani, A. Chowdhary, S. Pisharody, D. Lu, and D. Huang, "A defense system for defeating DDoS attacks in SDN based networks," in *Proc. 15th ACM Int. Symp. Mobility Manage. Wireless Access*. New York, NY, USA: Association for Computing Machinery, Nov. 2017, pp. 83–92.

[146] Q. Qin, K. Poularakis, and L. Tassiulas, "A learning approach with programmable data plane towards IoT security," in *Proc. IEEE 40th Int. Conf. Distrib. Comput. Syst. (ICDCS)*, Nov. 2020, pp. 410–420.

[147] A. Malik, R. de Frein, M. Al-Zeyadi, and J. Andreu-Perez, "Intelligent SDN traffic classification using deep learning: Deep-SDN," in *Proc. 2nd Int. Conf. Comput. Commun. Internet (ICCCI)*, Mali, Jun. 2020, pp. 184–189.

[148] Q. Niyaz, W. Sun, and A. Y. Javaid, "A deep learning based DDoS detection system in software-defined networking (SDN)," 2016, *arXiv:1611.07400*.

[149] P. Wang, F. Ye, X. Chen, and Y. Qian, "Datanet: Deep learning based encrypted network traffic classification in SDN home gateway," *IEEE Access*, vol. 6, pp. 55380–55391, 2018.

[150] Y. Liu, T. Zhi, M. Shen, L. Wang, Y. Li, and M. Wan, "Software-defined DDoS detection with information entropy analysis and optimized deep learning," *Future Gener. Comput. Syst.*, vol. 129, pp. 99–114, Apr. 2022.

[151] J. Malik, A. Akhunzada, I. Bibi, M. Imran, A. Musaddiq, and S. W. Kim, "Hybrid deep learning: An efficient reconnaissance and surveillance detection mechanism in SDN," *IEEE Access*, vol. 8, pp. 134695–134706, 2020.

[152] N. Ahuja, G. Singal, and D. Mukhopadhyay, "DLSDN: Deep learning for DDOS attack detection in software defined networking," in *Proc. 11th Int. Conf. Cloud Comput., Data Sci. Eng. (Confluence)*, Jan. 2021, pp. 683–688.

[153] M. S. Elsayed, N.-A. Le-Khac, S. Dev, and A. D. Jurcut, "DDoSNet: A deep-learning model for detecting network attacks," in *Proc. IEEE 21st Int. Symp. World Wireless, Mobile Multimedia Netw. (WoWMoM)*, Aug. 2020, pp. 391–396.

[154] H. Wang, X. Tan, S. Yuan, M. Li, J. Wu, and Q. Zheng, "A two-phase encrypted traffic classification scheme in programmable data plane," in *Proc. IEEE Int. Symp. Parallel Distrib. Process. Appl. (ISPA)*, Oct. 2024, pp. 2272–2273.

[155] R. Ye, Y. Ouyang, and X. Che, "Security and attack prevention in software-defined network," in *Proc. Int. Conf. Telecommun. Power Electron. (TELEPE)*, May 2024, pp. 824–828.

[156] M. A. Kumar, A. H. Pai, J. Agarwal, S. Christa, G. M. S. Prasad, and S. Saifi, "Deep learning model to defend against covert channel attacks in the SDN networks," in *Proc. Adv. Comput. Commun. Technol. High Perform. Appl. (ACCTHPA)*, Jan. 2023, pp. 1–5.

[157] A. Alkhamisi, I. Katib, and S. M. Buhari, "Federated learning-based security attack detection for multi-controller software-defined networks," *Algorithms*, vol. 17, no. 7, p. 290, Jul. 2024.

[158] A. Sebbar, O. Cherqi, K. Chougdali, and M. Boulmalf, "Real-time anomaly detection in SDN architecture using integrated SIEM and machine learning for enhancing network security," in *Proc. GLOBECOM-IEEE Global Commun. Conf.*, Dec. 2023, pp. 1795–1800.

[159] M. Seufert, K. Dietz, N. Wehner, S. Geißler, J. Schüler, M. Wolz, A. Hotho, P. Casas, T. Hoßfeld, and A. Feldmann, "Marina: Realizing ML-driven real-time network traffic monitoring at terabit scale," *IEEE Trans. Netw. Service Manage.*, vol. 21, no. 3, pp. 2773–2790, Jun. 2024.

[160] P. Golchin, C. Zhou, P. Agnihotri, P. Agnihotri, M. Hajizadeh, R. Kundel, and R. Steinmetz, "CML-IDS: Enhancing intrusion detection in SDN through collaborative machine learning," in *Proc. 19th Int. Conf. Netw. Service Manage. (CNSM)*, Oct. 2023, pp. 1–9.

[161] G. Zhou, Z. Liu, C. Fu, Q. Li, and K. Xu, "An efficient design of intelligent network data plane," in *Proc. 32nd USENIX Secur. Symp. (USENIX Secur.)*, Aug. 2023, pp. 6203–6220.

[162] A. Nascimento, D. Abreu, A. Riker, and A. Abelém, "AID-SDN: Advanced intelligent defense for SDN using P4 and machine learning," in *Proc. IEEE Latin-Amer. Conf. Commun. (LATINCOM)*, Nov. 2023, pp. 1–6.

[163] N. Maheswaran, S. Bose, G. Gokulraj, T. Anitha, T. Shruthi, and G. Vijayaraj, "Intrusion prevention system in SDN environment for 6G networks using deep learning," in *Proc. 6th Int. Conf. Mobile Comput. Sustain. Informat. (ICMCSI)*, Jan. 2025, pp. 53–61.

[164] M. G. Spina, F. De Rango, E. Scalzo, F. Guerriero, and A. Iera, "Distributing intelligence in 6G programmable data planes for effective in-network intrusion prevention," *IEEE Netw.*, vol. 39, no. 3, pp. 319–325, May 2025.

[165] E. Bardhi, M. Conti, and R. Lazzeretti, "Is AI a trick or T(h)reat for securing programmable data planes?" *IEEE Netw.*, vol. 38, no. 6, pp. 146–152, Nov. 2024.

[166] R. H. Serag, M. S. Abdalzaher, H. A. E. A. Elsayed, M. Sobh, M. Krichen, and M. M. Salim, "Machine-Learning-Based traffic classification in software-defined networks," *Electronics*, vol. 13, no. 6, p. 1108, Mar. 2024.

[167] F. Paolucci, F. Cugini, P. Castoldi, and T. Osinski, "Enhancing 5G SDN/NFV edge with P4 data plane programmability," *IEEE Netw.*, vol. 35, no. 3, pp. 154–160, May 2021.

[168] O. Michel, R. Bifulco, G. Rétvári, and S. Schmid, "The programmable data plane: Abstractions, architectures, algorithms, and applications," *ACM Comput. Surv.*, vol. 54, no. 4, pp. 1–36, May 2021.

[169] E. Kaljic, A. Maric, P. Njemcevic, and M. Hadzialic, "A survey on data plane flexibility and programmability in software-defined networking," *IEEE Access*, vol. 7, pp. 47804–47840, 2019.

[170] A. AlSabeh, J. Khoury, E. Kfoury, J. Crichigno, and E. Bou-Harb, "A survey on security applications of P4 programmable switches and a STRIDE-based vulnerability assessment," *Comput. Netw.*, vol. 207, Apr. 2022, Art. no. 108800.

[171] E. F. Kfoury, J. Crichigno, and E. Bou-Harb, "An exhaustive survey on P4 programmable data plane switches: Taxonomy, applications, challenges, and future trends," *IEEE Access*, vol. 9, pp. 87094–87155, 2021.

[172] Y. Gao and Z. Wang, "A review of P4 programmable data planes for network security," *Mobile Inf. Syst.*, vol. 2021, pp. 1–24, Nov. 2021.

[173] B. Goswami, M. Kulkarni, and J. Paulose, "A survey on P4 challenges in software defined networks: P4 programming," *IEEE Access*, vol. 11, pp. 54373–54387, 2023.

[174] S. Demirci and S. Sagiroglu, "Optimal placement of virtual network functions in software defined networks: A survey," *J. Netw. Comput. Appl.*, vol. 147, Dec. 2019, Art. no. 102424.

[175] S. Demirci, M. Demirci, and S. Sagiroglu, "Virtual security functions and their placement in software defined networks: A survey," *Gazi Univ. J. Sci.*, vol. 32, no. 3, pp. 833–851, Sep. 2019.

[176] D. Moro, G. Verticale, and A. Capone, "Network function decomposition and offloading on heterogeneous networks with programmable data planes," *IEEE Open J. Commun. Soc.*, vol. 2, pp. 1874–1885, 2021.

[177] M. Mehic, M. Niemiec, S. Rass, J. Ma, M. Peev, A. Aguado, V. Martin, S. Schauer, A. Poppe, C. Pacher, and M. Voznak, "Quantum key distribution: A networking perspective," *ACM Comput. Surv.*, vol. 53, no. 5, pp. 1–41, May 2020.

[178] P. Porambage, G. Gür, D. P. M. Osorio, M. Livanage, and M. Ylianttila, "6G security challenges and potential solutions," in *Proc. Joint Eur. Conf. Netw. Commun. 6G Summit (EuCNC/6G Summit)*, Jun. 2021, pp. 622–627.

[179] M. Brauer, R. J. Vicente, J. S. Buruaga, R. B. Méndez, R.-P. Braun, M. Geitz, P. Rydlichkowski, H. H. Brunner, F. Fung, M. Peev, A. Pastor, D. R. Lopez, V. Martin, and J. P. Brito, "Linking QKD testbeds across Europe," *Entropy*, vol. 26, no. 2, p. 123, Jan. 2024.

[180] *Quantum Key Distribution (QKD); Application Interface*, document TSI GS QKD 004, 2020.

[181] *Quantum Key Distribution (QKD); Protocol and Data Format of REST-based Key Delivery API*, document ETSI GS QKD 014, 2019.

[182] *Quantum Key Distribution (QKD); Control Interface for Software Defined Networks*, document ETSI GS QKD 015, 2022.

[183] R. S. Tessinari, R. I. Woodward, and A. J. Shields, "Software-defined quantum network using a QKD-secured SDN controller and encrypted messages," in *Proc. Opt. Fiber Commun. Conf. Exhib. (OFC)*, Mar. 2023, pp. 1–3.

[184] D. R. Lopez, V. Martin, V. Lopez, F. de la Iglesia, A. Pastor, H. Brunner, A. Aguado, S. Bettelli, F. Fung, D. Hillerkuss, L. Comandar, D. Wang, A. Poppe, J. P. Brito, P. J. Salas, and M. Peev, "Demonstration of software defined network services utilizing quantum key distribution fully integrated with standard telecommunication network," *Quantum Rep.*, vol. 2, no. 3, pp. 453–458, Sep. 2020.

[185] A. Aguado, V. Lopez, D. Lopez, M. Peev, A. Poppe, A. Pastor, J. Folgueira, and V. Martin, "The engineering of software-defined quantum key distribution networks," *IEEE Commun. Mag.*, vol. 57, no. 7, pp. 20–26, Jul. 2019.

[186] V. Martin et al., "MadQCI: A heterogeneous and scalable SDN QKD network deployed in production facilities," 2023, *arXiv:2311.12791*.
[187] V. Lopez, A. Pastor, D. Lopez, A. Aguado, and V. Martin, "Applying QKD to improve next-generation network infrastructures," in *Proc. Eur. Conf. Netw. Commun. (EuCNC)*, Jun. 2019, pp. 283–288.
[188] A. Aguado, E. Hugues-Salas, P. A. Haigh, J. Marhuenda, A. B. Price, P. Sibson, J. E. Kennard, C. Erven, J. G. Rarity, M. G. Thompson, A. Lord, R. Nejabati, and D. Simeonidou, "First experimental demonstration of secure NFV orchestration over an SDN-controlled optical network with time-shared quantum key distribution," in *Proc. 42nd Eur. Conf. Opt. Commun.*, Sep. 2016, pp. 1–3.
[189] A. Aguado, E. Hugues-Salas, P. A. Haigh, J. Marhuenda, A. B. Price, P. Sibson, J. E. Kennard, C. Erven, J. G. Rarity, M. G. Thompson, A. Lord, R. Nejabati, and D. Simeonidou, "Secure NFV orchestration over an SDN-controlled optical network with time-shared quantum key distribution resources," *J. Lightw. Technol.*, vol. 35, no. 8, pp. 1357–1362, Apr. 15, 2017.
[190] C. Bu, X. Wang, H. Cheng, M. Huang, K. Li, and S. K. Das, "Enabling adaptive routing service customization via the integration of SDN and NFV," *J. Netw. Comput. Appl.*, vol. 93, pp. 123–136, Sep. 2017.
[191] B. Isyaku, K. B. A. Bakar, F. A. Ghaleb, and A. Al-Nahari, "Dynamic routing and failure recovery approaches for efficient resource utilization in OpenFlow-SDN: A survey," *IEEE Access*, vol. 10, pp. 121791–121815, 2022.
[192] J. Ali, G.-M. Lee, B.-H. Roh, D. K. Ryu, and G. Park, "Software-defined networking approaches for link failure recovery: A survey," *Sustainability*, vol. 12, no. 10, p. 4255, May 2020.
[193] Y. Zhang, L. Qiu, Y. Xu, X. Wang, S. Wang, A. Paul, and Z. Wu, "Multi-path routing algorithm based on deep reinforcement learning for SDN," *Appl. Sci.*, vol. 13, no. 22, p. 12520, Nov. 2023.
[194] X. Guo, H. Lin, Z. Li, and M. Peng, "Deep-Reinforcement-Learning-Based QoS-aware secure routing for SDN-IoT," *IEEE Internet Things J.*, vol. 7, no. 7, pp. 6242–6251, Jul. 2020.
[195] Z. Li, Y. Hu, J. Wu, and J. Lu, "P4Resilience: Scalable resilience for multi-failure recovery in SDN with programmable data plane," *Comput. Netw.*, vol. 208, May 2022, Art. no. 108896.
[196] C. Cascone, D. Sanvito, L. Pollini, A. Capone, and B. Sanso, "Fast failure detection and recovery in SDN with stateful data plane," *Int. J. Netw. Manage.*, vol. 27, no. 2, p. 1957, Mar. 2017.
[197] M. Wang, J. Liu, J. Mao, H. Cheng, J. Chen, and C. Qi, "RouteGuardian: Constructing secure routing paths in software-defined networking," *Tsinghua Sci. Technol.*, vol. 22, no. 4, pp. 400–412, Aug. 2017.

**AMINA TANKOVIC** (Member, IEEE) received the bachelor's and master's degrees in telecommunication engineering from the Faculty of Electrical Engineering, University of Sarajevo, Bosnia and Herzegovina, in 2019 and 2021, respectively, where she is currently pursuing the Ph.D. degree. She is a Teaching Assistant. Her research interests include software-defined networking, network security, and quantum cryptography.

**EMIR DERVISEVIC** was born in Berlin, Germany, in 1995. He is currently pursuing the Ph.D. degree with the Department of Telecommunications, Faculty of Electrical Engineering, University of Sarajevo. Since 2020, he has been a part of the international scientific research projects Horizont 2020-Open European QuantumKey Distribution Testbed (OPENQKD) and NATO SPS MYP G5894-Quantum Cybersecurity in 5G Networks (QUANTUM5). He is actively developing the quantum key distribution network simulation module. His research interests include quantum key distribution networks, network management, network security, and cryptography (For more details: www.open-qkd.eu).

**MIRALEM MEHIC** (Member, IEEE) received the Ph.D. degree in telecommunications from the VSB—Technical University of Ostrava, Czechia. He also studied from the AGH University of Science and Technology, Krakow, Poland; Alpen-Adria-Universität Klagenfurt, Austria; and Austrian Institute of Technology (AIT) in the Department of Digital Safety and Security Business Units-Optical Quantum Technology, Vienna and Klagenfurt, Austria. Since 2019, he has been the Head of the Department of Telecommunications, University of Sarajevo, Bosnia and Herzegovina. He is the author of the Unique QKD Network Simulator QKDNETSIM. His research interests include quality of service and management of QKD networks with a focus on real-time traffic and the utilization of network resources. He is the national principal investigator of EU H2020 OPENQKD and the NATO SPS G5894 QUANTUM5 projects (For more details: www.qkdnetsim.info).

**ENIO KALJIC** received the Ph.D. degree in telecommunications from the Faculty of Electrical Engineering, University of Sarajevo, Bosnia and Herzegovina. He is currently an Associate Professor with the Department of Telecommunications. His research interests include software-defined networking, cybersecurity, and packet-switching architectures for wireless and wired networking.