



University of West Attica
School of Engineering
DEPARTMENT OF INFORMATION AND COMPUTER ENGINEERING

MASTER DISSERTATION

Security in Software Defined Networks

MICHALIS ANDROULAKAKIS
R.N cscy21004

Supervisor:
Ioanna Kantzavelou

Security in Software Defined Networks

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

Security in Software Defined Networks

MICHALIS ANDROULAKAKIS
R.N cscy21004

Supervisor:

Ioanna Kantzavelou, Assistant Professor

Examination Committee:

Leandros Maglaras, Professor



Panayiotis Yannakopoulos, Professor

Examination Date 09/02/2024

Security in Software Defined Networks

ΔΗΛΩΣΗ ΣΥΓΓΡΑΦΕΑ ΜΕΤΑΠΤΥΧΙΑΚΗΣ ΕΡΓΑΣΙΑΣ

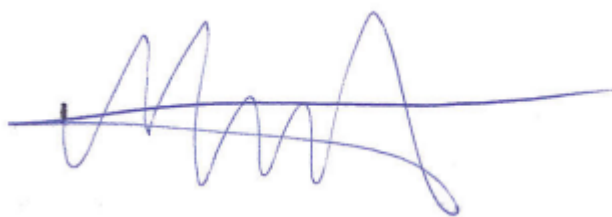
«Βεβαιώνω ότι είμαι συγγραφέας αυτής της Διπλωματικής εργασίας και κάθε βοήθεια την οποία είχα για την προετοιμασία της, είναι πλήρως αναγνωρισμένη και αναφέρεται στην εργασία. Επίσης, οι όποιες πηγές από τις οποίες έκανα χρήση δεδομένων, ιδεών ή λέξεων, είτε ακριβώς είτε παραφρασμένες, αναφέρονται στο σύνολό τους, με πλήρη αναφορά στους συγγραφείς, τον εκδοτικό οίκο ή το περιοδικό, συμπεριλαμβανομένων και των πηγών που ενδεχομένως χρησιμοποιήθηκαν από το διαδίκτυο. Επίσης, βεβαιώνω ότι αυτή η εργασία έχει συγγραφεί από μένα αποκλειστικά και αποτελεί προϊόν πνευματικής ιδιοκτησίας τόσο δικής μου, όσο και του Ιδρύματος.

Παράβαση της ανωτέρω ακαδημαϊκής μου ευθύνης αποτελεί ουσιώδη λόγο για την ανάκληση του πτυχίου μου».

«Βεβαιώνω ότι είμαι συγγραφέας της παρούσας διπλωματικής εργασίας και ότι έχω αναφέρει ή παραπέμψει σε αυτή, ρητά και συγκεκριμένα, όλες τις πηγές από τις οποίες έκανα χρήση δεδομένων, ιδεών, προτάσεων ή λέξεων, είτε αυτές μεταφέρονται επακριβώς (στο πρωτότυπο ή μεταφρασμένες) είτε παραφρασμένες. Επίσης βεβαιώνω ότι αυτή η εργασία προετοιμάστηκε από εμένα προσωπικά ειδικά για την συγκεκριμένη διπλωματική εργασία»

Ο/Η Δηλών/ούσα

MICHALIS ANDROULAKAKIS



ΕΥΧΑΡΙΣΤΙΕΣ

Η παρούσα διπλωματική εργασία ολοκληρώθηκε μετά από επίμονες προσπάθειες, σε ένα ενδιαφέρον γνωστικό αντικείμενο, όπως αυτό της επεξεργασίας κειμένου. Την προσπάθειά μου αυτή υποστήριξε η επιβλέπων καθηγήτρια Ιωάννα Καντζάβελου, την οποία θα ήθελα να ευχαριστήσω θερμά.

Abstract

Software-Defined Networking (SDN) represents a transformative shift in network architecture, offering centralized control and programmability. This thesis investigates SDN's architecture, its unique security challenges, and essential security principles. SDN-specific security concerns such as centralized control, programmability, legacy protocol integration, and cross-domain connectivity are explored. Security principles, including defining security dependencies, robust identity assurance, reliance on open standards, protection of the information security triad, safeguarding operational reference data, secure default configurations, accountability, and traceability, serve as guiding principles.

The thesis delves into various attack vectors, including data plane threats like man-in-the-middle attacks and BGP route hijacking, as well as application layer vulnerabilities such as Cross-Site Scripting (XSS), SQL Injection, and Cross-Site Request Forgery (CSRF) attacks.

The research underscores the necessity of transitioning to SDN and conducts a fundamental analysis of its architecture. Furthermore, it presents a comprehensive examination of the threat landscape and vulnerabilities associated with SDN. Finally, a holistic defense strategy for SDN is proposed, encompassing policies, distributed firewalls, multi-layer intrusion detection and protection systems (IDPS), artificial intelligence, and moving target defense mechanisms. The results offer valuable insights into securing SDN deployments and mitigating evolving threats in modern network environments.

Table of Contents

Abstract	7
Table of Contents	8
Chapter 1: SDN	10
1.1. Why a shift to software-defined networks is necessary	10
1.2. Architecture	12
1.3. SDN-Specific Security Challenges	22
1.3.1 Centralized Control	23
1.3.2 Programmability	24
1.3.3 Challenge of Integrating Legacy Protocols	25
1.3.4 Cross-Domain Connection	25
1.4. Security Principles	26
1.3.1 Principle 1: Clearly Defining Security Dependencies and Trust Boundaries	26
1.3.2 Principle 2: Assuring Robust Identity	26
1.3.3 Principle 3: Building Security Based on Open Standards	27
1.3.4 Principle 4: Protect the Information Security Triad	28
1.3.5 Principle 5: Protecting Operational Reference Data	29
1.3.6 Principle 6: Making Systems Secure by Default	29
1.3.7 Principle 7: Provide Accountability and Traceability	30
1.3.8 Principle 8: Properties of Manageable Security Controls	31
Chapter 2: Attacks on all layers of the SDN architecture	33
2.1 Attacks on the Data Layer	33
2.1.1 Man-in-the-middle attack (MitM)	33
2.1.2 BGP Route Hijacking Attacks	35
2.1.3 Sniffing and Eavesdropping Attacks	37
2.2 Attacks on the Application layer	39
2.2.1 Cross-Site Scripting (XSS) Attack	39
2.2.2 SQL Injection Attack	42
2.2.3 Cross-Site Request Forgery (CSRF) Attack	44
2.3 Attacks on the Control layer	46
2.3.1 DDOS attack to controller by exploiting session hijacking:	46
2.3.2 Link fabrication attack	48
2.3.3 Flow table overloading	50
Chapter 3: Holistic defense strategy of SDN deployed in 5 basic axes	53
3.1 Policies in SDN:	53
3.2 Distributed firewalls and microsegmentation	56
3.3 Multi-layer Intrusion detection and prevention Systems(IDPS):	60
3.4 Artificial Intelligence and its types	66
3.5 Moving Target Defense Mechanisms	69

Chapter 4: Security challenges arise from the deployment of SDN in IOT	73
4.1 General description	73
4.2 Manufacturer Usage Description (MUD) solution	74
4.3 Secure IOT through the combination of MUD with ML	76
Chapter 5: Conclusions and the future of the SDN security	78
Bibliography	81

Chapter 1: SDN

1.1. Why a shift to software-defined networks is necessary

The exponential increase of the digital data, the cloud services, the commercialization of Information Technology, the data driven approach and the number of devices (PC, mobiles, IOT, Servers) lead the global information community to change the classic client-server architecture to a better, more scalable server to server architecture.

The new east-west traffic (server to server) constitutes a new paramount turning point and defines a whole new era for the digital world. With the purpose of improving the east-west traffic , the community is developing huge cutting-edge data centers around the world.

The commercialization of Information Technology in conjunction with the Cloud services are absolutely essential for the modern society of people and firms. A tangible example is that the 9 out of 10 most financially strong and influential firms sell Cloud Services and 85% of the world population uses smartphone and Cloud services.

The new data driven approach is a way of improving the strategy and the results of every organization based on the data analysis and interpretation. Modern marketing is indisputably data driven. This can be done, only by gathering all the data from all the digital environments from an organization. This procedure requires massive computational power, storage and bandwidth.

At that point the wider Information Technology community understood that the conventional distributed architecture of networks, as it was developed many years ago, could not efficiently support the emerging technology of modern data centers, virtualization and all the above needs. Thus, a new centralized architecture started prospering the Software Defined Networking(SDN). At this point it will be cited as an economical diagram just to prove the importance and the magnitude of the adoption of the SDN emerging technology.

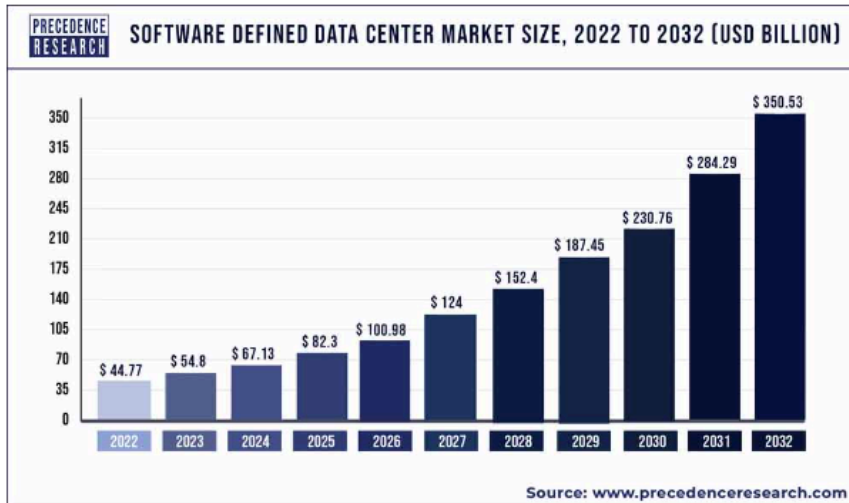


Figure 1, Growth of SDN in the coming years [39]

As it is stated in the picture above (Figure 1) the current market size at the moment is around 55 USD billions and it is expected to reach more than 350 USD billions before the next decade passes. The percentage of the Data Centers which will utilize SDN as a network solution will be almost 50% before 2033.

The SDN constitutes a very general term. SDN usages could be categorized in three basic categories, the SDN which support SD-WAN solutions, the SDN solutions which can support SD-LAN solutions and even beyond in the SDN tools which are dedicated to support Data Centers and especially Application-Centric solutions: Some very known SDN tools used globally by big firms.

Cisco's ACI (Application Centric Infrastructure) creates a powerful, adaptive and flexible network and can be used as a scalable solution for data centers in which virtual machines and containers are hosted. Also, ACI is employed to develop web applications in data centers.

VMware NSX is a network virtualization and security platform that enables the creation and management of virtual networks. It's widely known and often used in conjunction with VMware's virtualization products. It should be noted that VMware has 46% of the virtualization technology market share globally.

SDA (Software Defined Access) is a centralized, controller based network solution and a totally new way to create an entire campus LAN.

SD-WAN (Software Defined Wide Area Network) is a centralized, controller based network solution, steering traffic securely and intelligently through the WAN and directly to trusted SaaS and IaaS providers.

It's worth mentioning that the development of SDN partially rests upon the NFV technology. SDN can be enhanced by the integration with NFV.

NFV (Network Functions Virtualization) provides network services such as intrusion and detection prevention systems, firewalls, load balancers and other network functions which are deployed in a virtualized environment instead of a physical dedicated appliance. NFV in conjunction with SDN can increase the performance, the flexibility and the efficiency of Data Centers. However, NFV does not seek to change existing protocols in comparison with SDN which works in a totally radical way than conventional networks.

During my research for this thesis, I realized that the overwhelming majority of articles about security in SDN, dedicate a whole section about the IOT security in SDN and this proves that generally IOT devices pose a real security challenge. Thus, I think it is correct to dedicate a section for analyzing the security in IOT.

Growing technologies needing the scalability of SDN to prosper

We should keep in mind that SDN architecture is employed in many different cutting edge areas of technology with huge prominence and impact for humanity. Beyond SDN usage in developing data centers, the LTE(long term evolution) and 5G infrastructure takes advantage of SDN technologies. As it has already proved, the integration between SDN and 5G adds great value to the global mobile network.

Other areas where SDN offers great value are the subcategories of Smart City concept, such as the IOT(internet of things), smart grids and power grids, CAVs(connected and autonomous vehicles), robotics and smart manufacturing and blockchain technology. The common denominator in all above categories, is the utilization of countless small interconnected devices or sensors with low computational power and no satisfactory implementation of by design and by default security principles.

1.2. Architecture

SDN, or Software Defined Networking, represents a networking architectural approach that facilitates the management and control of networks through software applications. With SDN, the behavior of the entire network and its associated devices is programmatically controlled in a centralized manner using software applications that leverage open APIs [1].

To gain a comprehensive understanding of software-defined networks, it is essential to grasp the distinct planes involved in networking [1]:

Data Plane:

The data plane encompasses all activities related to data packets transmitted by end-users. This plane is developed to handle with high speed, efficiency and accuracy the routing and forwarding functions. The most vital component of this layer is the flow tables including all the information that each network appliance needs in order to decide how to proceed with the routing of each packet. This encompasses functions such as:

- Forwarding data packets.
- Segmentation and reassembly of data.
- Replication of packets for multicast communication.

Control Plane:

The control plane is the brain and the soul of the whole architecture and governs activities necessary for orchestrating data plane operations but does not directly involve end-user data packets. In essence, it serves as the intellectual core of the network. Control plane activities include:

- Creating routing tables.
- Establishing policies for packet handling.

In summary, SDN is a network architecture that employs software applications and open APIs to centrally manage and control network behavior. The network is divided into the data plane, responsible for end-user data packet activities, and the control plane, responsible for coordinating these data plane operations and making critical network decisions.

Software Defined Networking (SDN)

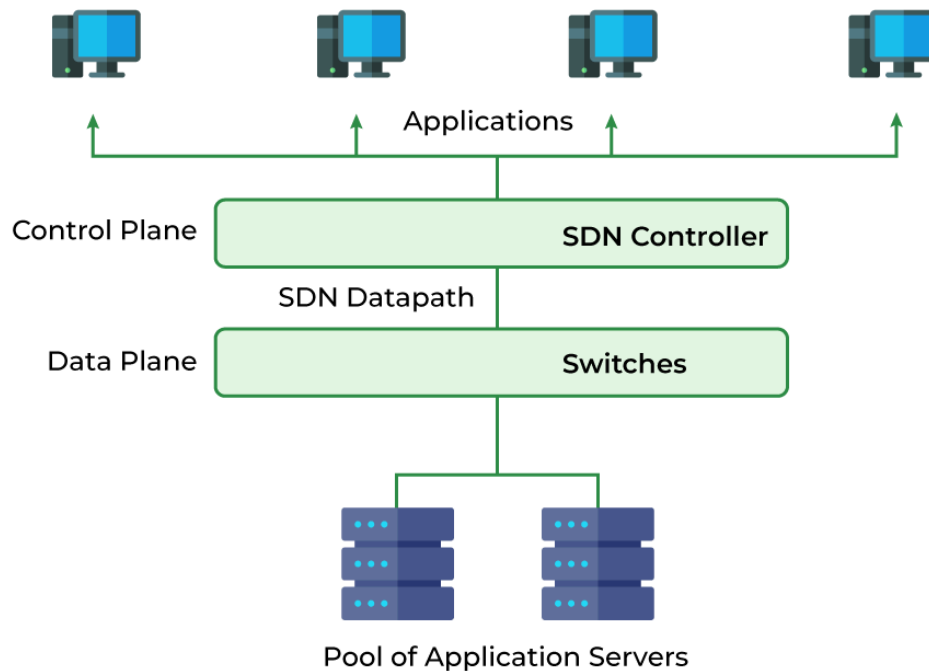


Figure 2, Application driven SDN architecture [40]

SDN, or Software Defined Networking, holds significant importance for several key reasons [2]:

1. Enhanced Network Connectivity:

SDN offers superior network connectivity, benefiting sales, services, and internal communications. It contributes to accelerated data sharing.

2. Streamlined Application Deployment (figure 2):

The deployment of new applications, services, and various business models can be expedited through the implementation of Software Defined Networking.

3. Heightened Security:

Software-defined networks provide heightened network visibility. Operators can establish distinct zones for devices requiring varying levels of security. SDN networks grant operators greater control.

4. Enhanced Control at High Speed:

Software-defined networking delivers superior speed compared to other networking paradigms, thanks to the utilization of an open-standard, software-based controller.

SDN, or Software Defined Networking, finds application in several contexts [3]:

1) Enterprise Deployment:

Enterprises widely employ SDN as a predominant approach for accelerating the deployment of applications while simultaneously reducing the associated deployment and operational expenses. SDN empowers IT administrators to centrally manage and provision network services.

2) Cloud Networking:

In the realm of cloud networking, SDN is leveraged in conjunction with white-box systems. Cloud service providers frequently opt for generic hardware, allowing for flexibility in configuring cloud data centers while realizing cost savings in terms of both capital expenditures (CAPEX) and operating expenditures (OPEX).

The fundamental constituents that constitute SDN consist of [4]:

A. SDN Applications:

SDN Applications serve as intermediaries that channel network requests or commands through the SDN Controller via application programming interfaces (APIs).

B. SDN Controller:

The SDN Controller functions as a pivotal component by aggregating network data from hardware elements and subsequently transmitting this information to SDN applications.

C. SDN Networking Devices:

SDN Networking Devices play a pivotal role in executing forwarding and data processing functions within the network infrastructure.

In a conventional network configuration, each switch operates with both its data plane and control plane functionalities. The control planes of various switches exchange information about the network's topology, ultimately constructing forwarding tables that dictate the appropriate routing of incoming data packets through the data plane.

SDN represents an approach where the control plane is extracted from individual switches and centralized within a unit known as the SDN controller. This centralization empowers network administrators to manage traffic patterns through a unified console, eliminating the need to interact with individual switches directly. The data plane, however, continues to reside within each switch. When a packet enters a switch, its routing is determined based on predefined flow table entries set by the controller [5].

A flow table comprises match fields, such as input port numbers and packet headers, along with corresponding instructions. When a packet arrives, it is matched against the entries in the flow table, and the associated instructions are executed. These instructions may involve forwarding the packet through one or more ports, discarding the packet, or appending headers to it. In cases where a packet does not find a matching entry in the flow table, the switch queries the SDN controller, which then dispatches a new flow entry to the switch. The switch subsequently handles the packet based on this updated flow entry [6].

A standard SDN architecture encompasses three layers (figure 3) [7]:

1. Application Layer:

The application layer encompasses customary network applications and functionalities utilized by organizations. These functionalities encompass intrusion detection systems, load balancing mechanisms, and firewalls. In contrast to a traditional network configuration that relies on dedicated appliances like firewalls and load balancers, a software-defined network adopts an alternative approach. It replaces these appliances with software applications that are orchestrated by a controller to govern the behavior of the data plane.

2. Control Layer:

The control layer serves as the central component in a Software-Defined Network, comprising the SDN controller software, which functions as the network's intellectual core. Situated on a server infrastructure, this controller assumes the responsibility of overseeing network policies and regulating the flow of traffic across the entire network.

3. Infrastructure Layer:

The infrastructure layer comprises physical switches that constitute the data plane, responsible for the actual movement of data packets.

Communication between these layers occurs through a set of interfaces known as north-bound APIs (connecting the application and control layers) and southbound APIs (linking the control and infrastructure layers).

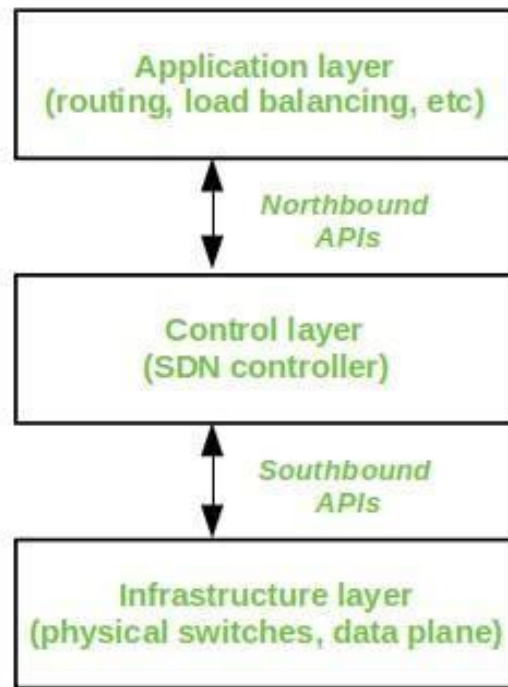


Figure 3, The three fundamental SDN layers [40]

Software-Defined Networking encompasses a range of technologies, including functional separation, network virtualization, and automation via programmability. Initially, SDN technology primarily focused on segregating the network's control plane from its data plane. In this context, the control plane governs packet flow decisions within the network, while the data plane is responsible for physically moving packets [8].

In a traditional SDN scenario, when a packet arrives at a network switch, the switch relies on predefined rules embedded in its proprietary firmware to determine how to route the packet. These rules are communicated to the switch by a centralized controller. The switch, also referred to as a data plane device, interacts with the controller as needed, providing information about the traffic it handles. It consistently

forwards all packets destined for the same endpoint along the same path, treating all packets uniformly [9].

Software-defined networking introduces an operational mode sometimes described as adaptive or dynamic. In this mode, a switch, when faced with a packet lacking a specific route, initiates a route request to the controller. It's important to note that this process differs from adaptive routing, which issues route requests through routers and algorithms based on network topology rather than through a central controller [10].

The virtualization aspect of SDN is realized through a virtual overlay, essentially a logically distinct network layered at the top of the physical infrastructure. This overlay empowers users to create end-to-end abstractions that obscure the underlying network, facilitating the segmentation of network traffic. This microsegmentation proves particularly advantageous for service providers and operators managing multi-tenant cloud environments and services, as it enables the provisioning of individual virtual networks with tailored policies for each tenant [11].

There are several models employed within the domain of SDN, each offering unique approaches [1]:

Open SDN (figure 4):

Open SDN is implemented by incorporating OpenFlow switches, representing a straightforward application of SDN principles. Within the Open SDN framework, the controller communicates with switches using a southbound API, facilitated by the OpenFlow protocol.

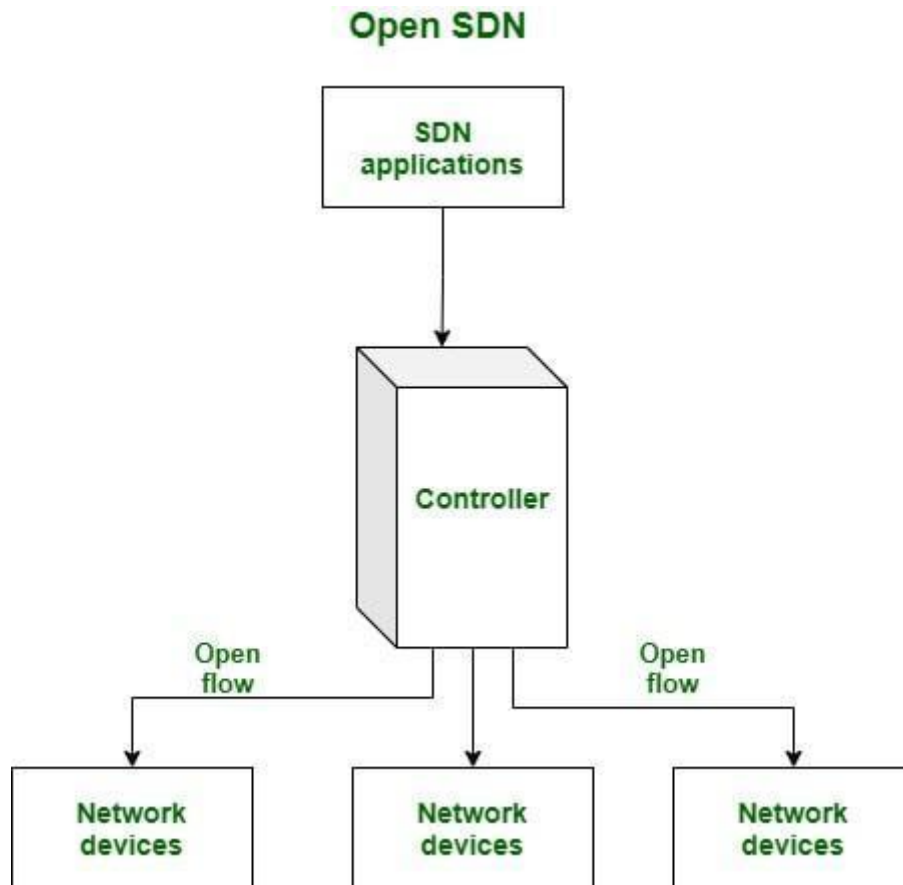


Figure 4, Depiction of Open SDN framework [40]

SDN via APIs:

In the SDN via API model, functions within remote devices such as switches are invoked using conventional methods like SNMP or CLI, as well as newer methods like Rest API. Here, control points are provided to devices, enabling the controller to manipulate remote devices through APIs.

SDN via Hypervisor-based Overlay Network (figure 5):

In the SDN via the hypervisor approach, the configuration of physical devices remains unchanged. Instead, hypervisor-based overlay networks are established atop the physical network. Only the devices situated at the periphery of the physical network are connected to these virtualized networks, thereby obscuring the details of other devices within the physical network.

SDN via Hypervisor

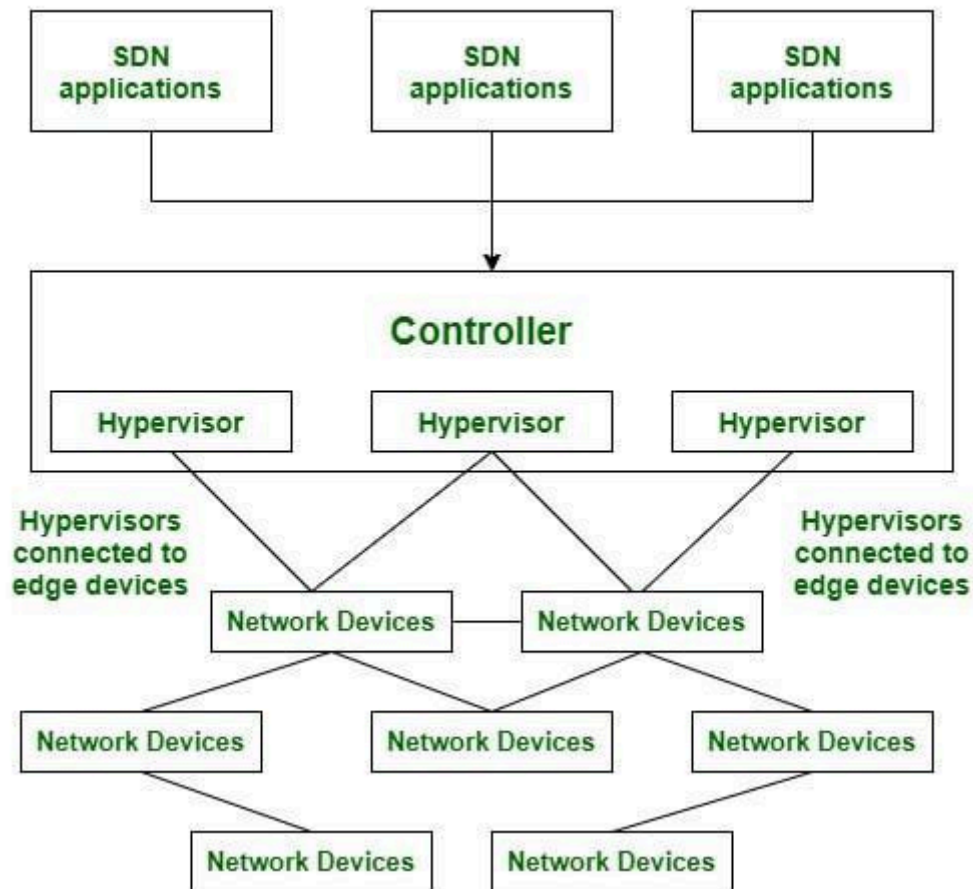


Figure 5, Depiction of SDN Hypervisor-based Overlay Network [40]

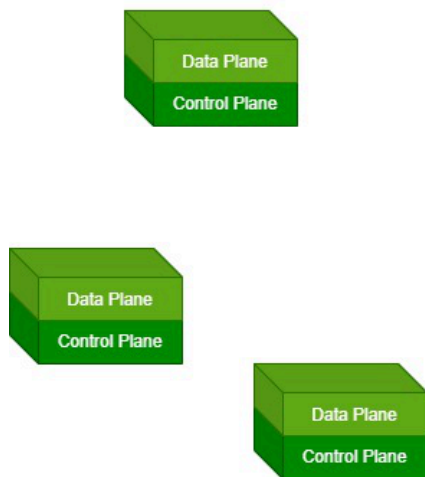
Hybrid SDN:

Hybrid SDN represents a networking approach that combines elements of both traditional networking and software-defined networking within a single network infrastructure. This amalgamation allows for the support of various types of network functions, catering to diverse operational requirements and functionalities.

Software Defined Networking represents a virtualized networking approach characterized by its centralized control, programmability, and open interfaces. In SDN, the data plane and control plane are decoupled through software, enabling flexible and dynamic network management (figure 6). On the other hand, Traditional Networking follows the conventional networking paradigm with distributed control mechanisms, lacking programmability and often relying on closed interfaces. In traditional networking, the data plane and control plane remain tightly integrated on

the same plane. These distinctions underline the fundamental differences between the two networking approaches, with SDN offering greater agility and control over network resources [12]. In the conventional networking architecture the network devices, the control and the data plane takes place inside the network device. Specifically the control and management planes run in the Internetwork Operating System through RAM and CPU while the data plane runs on the ASIC logic and the TCAM table.

Traditional Network



Software Defined Network

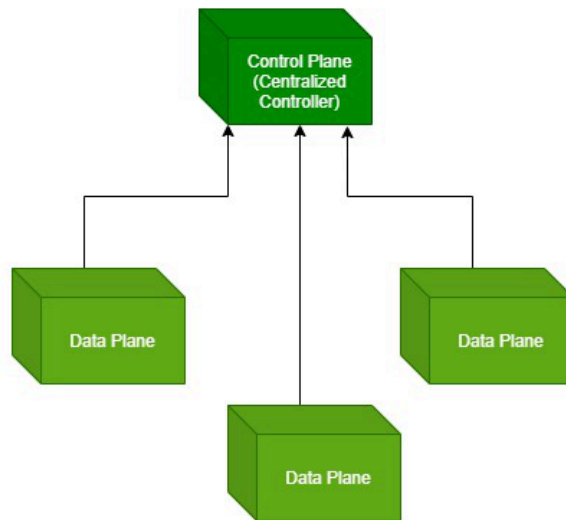


Figure 6, Differences between traditional networks and software defined networks [40]

Conclusion for SDN architecture:

We should take into consideration that the SDN architecture is not so strictly defined. The network architects can adjust as desiring the configurations depending on his goals. Many articles (the minority) do not even recognize the application plane in the SDN basic layers, and it is mentioned that there are only 2 layers, the data and control layers. The only common denominator of all resources is that the routing network appliances are managed in a centralized way from a controller living in the control layer, namely decoupling the control of traffic from routing . For example an SDN

architect can combine many different controllers in the same control layer and its custom developed APIs in order to connect all planes. If we add a controller in a conventional network and make him the orchestrator of the routing tables, then the network traffic is centralized and considered as an SDN solution.

1.3. SDN-Specific Security Challenges

The introduction of new features and the deployment of novel network configurations can introduce vulnerabilities and risks that create opportunities for threats previously nonexistent or more severe than before. In a specific network configuration described in reference [13], an SDN controller from one provider is able to interact directly with and control SDN switches owned by another provider. It's crucial to emphasize that implementing such a setup is highly discouraged. Additionally, aside from the usual methods of attacks targeting data flows, switches, administrative systems, and recovery/fault diagnosis procedures, the introduction of controllers and their communication in the Controller plane introduces fresh security issues that are specific to Software Defined Networking. Generally speaking SDN has by far more complex architecture than conventional networks and at the same time is a quite new concept supported by cutting edge technology. Given the rapid progress of SDN and its basic role in the development of infrastructure that supports data centers, SD-LAN and SD-WAN attackers have dedicated many time and effort in order to find exploitable vulnerabilities. The combination of the three factors above makes the SDN a quite attractive target for the attackers.

Understanding and addressing security in SDN is crucial to ensure the integrity, confidentiality, and availability of network resources and data in the three different planes(Application, Control, Data). We should keep in mind that analyzing the security of an SDN network it's quite a complex task because Application, Control and Data planes according to needs can be deployed in both virtual and physical servers. In a similar way, the Data plane can work virtually with OVS(Open Virtual Switch running virtually) [13] or with physical network appliances L2/L3 switches that support Openflow. Thus, it makes sense to analyze the security in each layer in correlation with the deployment environment. Different vulnerabilities and exploitable points arise in an SDN infrastructure whether the data layer is deployed on

virtual L2/L3 switches or on physical hardware appliances. It should be pointed out that in large networks(data centers, SD-Wan) the gateways or the focal points which control the ingress and the egress of the network are based on physical network appliances because they still can offer better performance and lower latency than virtual machines.

1.3.1 Centralized Control

Centralized control, sometimes referred to as logically centralized control (signifying distributed but coordinated control functions), presents a valuable target that could be vulnerable to potential attackers. A compromised controller can lead to unauthorized control over the entire network, potentially causing disrupting damage. If the SDN contains only one controller then availability means a single point of failure. These malicious individuals might attempt to influence essential network services or gain control over the entire network by deceiving or compromising a controller. This scenario differs from a scenario with numerous independent assets operating within a completely distributed control domain, as described in reference [14].

Assuring availability of the controller: One of the biggest challenges in SDN is the availability of the controllers, especially in a large demanding network. Controllers can be targeted from DDOS coming from both Application and Data layers. The most common queries are how many controllers do need and where is the best place to deploy them. Finally in a network with n controllers how many controllers do I need to support the back-up line and how do they cooperate. There are many different models and approaches of how to develop and deploy a strategy for primary and backup controllers. The most dominant approach for large scale networks with N controllers is the VBD (Virtual Backup Domain) approach. In VBD approach there are virtually created K clusters and separated into domains consisting of controllers (always $K < N$) [23]. Each VBD contains at least one dedicated backup controller. When a controller faces a failure then the respective VMD acts as backup controller. The merge of domains to VBD is based on the minimum average response time between the nodes. The problem of how controllers should function in a large scale network is a NP hard problem and to solve it are used game theory principles and

heuristic algorithms which work with different metrics. The majority of them operate with metrics based on latency and predefined topology.

1.3.2 Programmability

The direct programmatic access that SDN offers to clients, often distinct organizations or business entities, introduces novel security challenges. This evolving business model brings about security demands that are distinct from those found in closed administrative domains, especially concerning the protection of system integrity, third-party data, and open interfaces, as outlined in reference [14].

1.3.3.1 Traffic and Resource Isolation

It is imperative for operators to guarantee the thorough separation of business management and real-time control data for each entity from all other entities. The potential benefits of implementing established best practices from automated interfaces between customer and provider business support systems should be considered. This element pertains to the mitigation of the recognized security issue of multi-tenant traffic and resource isolation, thereby preventing any potential interference and misuse. The novel business model of Software-Defined Networking (SDN) introduces additional dynamic interactions that require further isolation to accommodate a variety of Service Level Agreements (SLAs), private addressing requirements, and other related considerations [14].

1.3.3.2 Trust Between Third-Party Applications and the Controller

The utilization of programmability in technology systems provides a range of possibilities for the development of market-oriented applications. However, this also introduces potential risks, as it exposes these systems to potential threats from malicious or inadequately protected applications. Enforcing authentication and implementing different levels of authorization during the application registration

process with the controller is of utmost importance in order to mitigate potential risks and minimize the controller's exposure [14].

1.3.3.3 Interface Security Protection on A-CPI and I-CPI

In addition to the utilization of Application-CPIs (A-CPIs) for communication with applications, a controller may also be subjected to control by a higher-level controller or engage in collaboration with another controller within the same hierarchical level. The lack of security measures implemented on these interfaces could make the Software-Defined Networking system vulnerable to potential malicious attacks. Hence, it is imperative to establish clear definitions of security attributes and operational checkpoints in order to ensure the security of both A-CPIs and I-CPIs [14].

1.3.3 Challenge of Integrating Legacy Protocols

The development of SDN interfaces and protocols takes place in an environment marked by escalating exploitation of technical and process vulnerabilities, with potentially severe consequences for security. However, past experiences have highlighted the complexity of retrofitting security features into existing technologies, with examples such as the Domain Name Server (DNS) and Border Gateway Protocol (BGP) illustrating this challenge. It is imperative to thoroughly assess compatibility before implementing legacy protocols, such as Network Address Translation (NAT) and BGP, into SDN. Furthermore, it is essential to avoid replicating or exacerbating weaknesses that may have been previously addressed within legacy architectural frameworks during the construction of the SDN framework [14].

1.3.4 Cross-Domain Connection

Another imperative in SDN implementation involves the capability to interconnect infrastructure from different domains. This can be achieved by linking controllers

from various providers via the Intermediate-Controller Plane Interface (I-CPI). Establishing mechanisms for trust relationships, authorization level determination to prevent misuse, and secure channel setup should all be carefully considered in this context [14].

1.4. Security Principles

1.3.1 Principle 1: Clearly Defining Security Dependencies and Trust Boundaries

In the establishment of security measures for SDN networks, it is imperative to meticulously define the security dependencies among various components while avoiding circular dependencies. This involves explicitly delineating trust boundaries, which, in turn, facilitates targeted risk assessment and evaluation of security controls. Trust boundaries should be established by identifying the specific locations where shifts in privilege occur, the movement of information between different domains (such as ingress and egress directions), and the reliance on data that cannot be sufficiently verified in terms of confidentiality and integrity. It is advisable to regard any external dependency as a boundary of trust, as it is rational to anticipate potential threats originating from external systems. As a result, it is imperative that interfaces to external environments integrate robust security functionalities in order to effectively prevent or mitigate externally initiated attacks. To mitigate system risk, it is advisable to employ a least-privilege approach in order to restrict access from external systems. Furthermore, it is imperative to establish and enforce measures aimed at effectively managing or containing internally initiated attacks in order to mitigate their potential impact on the external environment [14].

1.3.2 Principle 2: Assuring Robust Identity

The fundamental basis for ensuring robust security lies in the capacity to distinctly identify all elements and users within a system, while also validating their identities through reliable and trusted sources. The absence of a robust identity framework

hinders the ability to establish comprehensive systems for authentication, authorization, and accounting. A comprehensive identity framework should exhibit the following characteristics [14]:

- The capacity to distinguish its possessor from other entities within a predetermined scope.
- The ability to be created, modified, and invalidated.

The primary objective is to mitigate the occurrence of impersonation, with a preference for employing resilient cryptographic mechanisms.

The examination of SDN architecture brings attention to possible pathways for components within the system's trust boundary to undermine the accessibility of the logically centralized control. Therefore, the robustness of the system is heavily reliant on a secure authentication process that is firmly grounded in verified identity. There are multiple instances in which external entities, such as network applications, necessitate access to particular system resources through established interfaces. In the given situations, it is recommended to utilize access control mechanisms that incorporate varying levels of privileges to grant authorization to external entities and verify their access to the system. One such example is role-based access control [9]. In the context of communication, a device's identity can be made known in two ways: explicitly or implicitly. Explicit indication involves the transmission of information such as identifiers, credentials, and IP addresses along with the packets. On the other hand, implicit indication occurs through the use of a cryptographic key to secure the packets [14].

1.3.3 Principle 3: Building Security Based on Open Standards

The utilization of open standards provides benefits in relation to both the transportability and compatibility of systems. When possible, it is recommended to employ established protocols and methodologies instead of developing new ones. The consideration of developing new protocols and algorithms should be reserved as a final option, to be pursued only when existing requirements cannot be sufficiently fulfilled. To ensure the security of the OpenFlow™ communication channel, it is crucial to implement transport layer protection for the Transmission Control Protocol (TCP) traffic headers and payloads involved. A multitude of TCP enhancement techniques have been suggested and extensively implemented for this objective [15].

Therefore, it is advisable to embrace established techniques instead of pursuing innovative solutions for the transport layer. The concept of reusing protocols and algorithms holds significant importance, particularly within the realm of security functionalities like encryption, authentication, and integrity. These functionalities require thorough evaluation to guarantee their resilience. It is important to acknowledge that the utilization of outdated protocols or algorithms, such as MD5 and Transport Layer Security (TLS) 1.0, which have been proven to be vulnerable and are no longer supported by recognized standards organizations, should be abstained from [14].

1.3.4 Principle 4: Protect the Information Security Triad

The primary objective of security controls is to augment the confidentiality, integrity, and availability (CIA) of a system. However, it is crucial to assess their influence on the overall security posture of the architecture. One crucial aspect of evaluating the efficacy of new controls is to assess the potential unintended consequence of diminishing the overall availability of the system. These controls should not introduce new vulnerabilities or opportunities for exploitation. It is imperative to acknowledge and address any reduction in the efficacy of the fundamental principles encapsulated by the CIA triad. When incorporating a centralized security server into the SDN architecture, it is imperative to conduct a thorough evaluation to determine its vulnerability to denial-of-service (DoS) attacks and the potential impact on system availability. If there are any potential risks, it is necessary to identify suitable measures to mitigate them. Furthermore, it is imperative to ensure that security controls are devised in a manner that minimizes any unwarranted decline in system performance and the introduction of excessive system intricacy, as this may potentially give rise to new security vulnerabilities. The selection of a security control is determined by a combination of security requirements, cost factors, and manageability considerations, as observed in practical scenarios [14].

1.3.5 Principle 5: Protecting Operational Reference Data

The effectiveness of a security control is intrinsically connected to the reliability of operational reference data, which includes important components like credentials and sequence numbers. These elements are crucial in making well-informed operational choices. The presence of erroneous reference data has the potential to cause unanticipated system behavior, which may ultimately lead to compromises in the areas of confidentiality, integrity, and/or availability. In addition, the unintentional exposure of confidential reference data, such as cryptographic keys, can result in security breaches of the security control system. Hence, it is imperative to explicitly establish and protect operational reference data for all security controls at a level of continuity that aligns with the current security policy and assumptions within the security architecture [14].

The generation, processing, maintenance, and transportation of reference data should be conducted in a secure manner, both during expected operational states and state transitions, as well as throughout the entire lifecycle of the system. This encompasses various states, including system initialization, standard operation, standby operation, failover conditions, system recovery, and the transitions between these states. An illustration of this concept can be observed in various security protocols, where the detection of replay attacks is facilitated through the utilization of monotonically increasing sequence numbers. Therefore, it is imperative to diligently prevent any uncontrolled rollback of these numbers, particularly in the event of system failures. The importance of this aspect becomes particularly significant in situations where there is a lack of support for automated key management [14].

1.3.6 Principle 6: Making Systems Secure by Default

The implementation of security controls should include the provision of various security levels that can accommodate the diverse requirements of different system use cases. These security levels should range from a disabled state to one that aligns with the most stringent security requirements, such as a default denial state. Regardless of the intended application, it is imperative for the system to establish a fundamental level where the majority of primary security controls are automatically enabled. Moreover, it is imperative that these controls are configured in a manner that aligns

with the minimum criteria, thus guaranteeing their effectiveness. It is imperative that security controls have the capacity to be reconfigured or disabled; however, such actions should only be undertaken following a deliberate determination made by the system owner or operator [14].

When implementing an authentication control, it is crucial to ensure that a default form of authentication is in place. In order to ensure the efficacy of control measures, it is imperative that authentication is not set to a null value or completely disabled. Similarly, it is imperative to ensure that the fundamental security characteristics of a system, which may differ depending on the specific circumstances, are consistently upheld during updates, recovery from failures, system restarts, and comparable scenarios [14].

1.3.7 Principle 7: Provide Accountability and Traceability

It is imperative that all security controls possess the capability to undergo auditing, with a specific emphasis on critical states and actions that are essential to the overall security of the system. The data that is recorded should possess an adequate amount of information that is suitable for the purpose of conducting an audit. Based on the available recorded data, an auditor possesses the ability to not only distinctly identify the entity accountable for a particular action but also to systematically trace the corresponding sequence of said action. The second principle is involved in the process of attributing actions to particular entities. Nevertheless, it is of equal importance to guarantee that the audited data remains free from unnecessary information and that the auditing procedure itself adheres to security policies without any violations. It is imperative to ensure that the security attributes of logged data are protected consistently with the security policy and underlying assumptions throughout its lifecycle. In order to ensure the security and integrity of the data, it is imperative to implement measures that protect against unauthorized access and unauthorized modifications [14].

1.3.8 Principle 8: Properties of Manageable Security Controls

In addition to the aforementioned seven principles, it is important to consider several properties of the control when introducing new controls into an architecture or standard [14]:

- In order to effectively design and implement a security control, it is crucial to clearly define the security objectives and underlying assumptions.
- Scalability is an important aspect to consider when implementing security controls. It is crucial that these controls are designed in a manner that allows for seamless integration across a wide range of installations, varying from the smallest reference system to the largest deployment. It is essential to ensure that this scalability is achieved without introducing unnecessary complexity.
- Impact Assessment: It is crucial to conduct an evaluation of the potential effects on solution implementation and lifecycle management when implementing new controls. The implementation of new security functions should prioritize minimal complexity, while also considering the importance of extensibility to accommodate the future integration of additional security control functions.
- The ease of implementing and maintaining security controls is crucial, as they should be uncomplicated to put into practice, sustain, and manage.
- Backward compatibility is an essential aspect of control design, as it ensures that controls are either compatible with previous versions or offer a means of upgrading to accommodate both existing and outdated controls.
- Documentation and adherence to standards are crucial aspects of security controls. It is imperative that security controls are thoroughly documented and established in accordance with clearly defined standards.
- Credential management involves the ability to revoke and modify security credentials as an integral aspect of a system's lifecycle.
- Supporting automation is important for security controls to ensure their proper implementation. Manual processes can sometimes result in incorrect configurations, which can decrease the effectiveness of the controls.

- The ability to monitor, troubleshoot, and debug a system is essential for its effective implementation.

Chapter 2: Attacks on all layers of the SDN architecture

2.1 Attacks on the Data Layer

2.1.1 Man-in-the-middle attack (MitM)

A Man-in-the-Middle (MiTM) attack is a form of cyber assault wherein the attacker covertly intercepts and mediates messages between two parties who believe they are engaging in direct communication. This attack method essentially constitutes a surreptitious form of eavesdropping, wherein the attacker intercepts and exerts control over the entire dialogue [16].

MiTM cyberattacks (figure 7) pose a significant and imminent threat to online security due to their capacity to capture and manipulate highly sensitive personal information in real-time. This includes critical data such as login credentials, account particulars, or credit card numbers [16].

These cyberattacks are sometimes referred to by various names, including "monster-in-the-middle," "machine-in-the-middle," "monkey-in-the-middle," and "man-in-the-browser" attacks. The "man-in-the-browser" variant is the most prevalent MiTM attack, where attackers focus on infecting web browsers and infuse malicious proxy malware into the victim's computing device. Typically, the introduction of this malware occurs through phishing emails. The primary objective underlying these attacks is the unlawful acquisition of financial information by intercepting a user's traffic when accessing banking or financial websites [16].

During MiTM attacks, cybercriminals position themselves between data transactions or online communications. By disseminating malware, the attacker gains unimpeded access to the user's web browser and the data transmitted and received during online transactions. MiTM attacks primarily target online banking and e-commerce platforms, which necessitate secure authentication involving public and private keys. These attacks enable malicious actors to capture login credentials and other confidential information [16].

Typically, these attacks follow a structured two-step process referred to as "data interception" and "decryption." Data interception involves the attacker intercepting a

data exchange occurring between a client and a server. In this scenario, the attacker employs deceptive tactics to make both the client and server believe they are engaged in a legitimate data exchange. Meanwhile, the attacker intercepts the data, establishes a connection with the genuine site, and serves as an intermediary to observe the communication while introducing fraudulent information [16].

The following sequence of actions outlines a prevalent data interception technique:

- i. The attacker deploys a packet sniffer, a tool designed to monitor network traffic for potential vulnerabilities, such as a user accessing a Hypertext Transfer Protocol (HTTP)-based website or utilizing an insecure public Wi-Fi hotspot.
- ii. Upon a user logging into the vulnerable website, the attacker seizes the user's data and redirects them to a counterfeit website.
- iii. The counterfeit website replicates the appearance and functionality of the original site, gathering all pertinent user information in the process. This information becomes accessible to the attacker, granting them potential access to the user's resources on the genuine website.

The decryption phase represents the stage where the intercepted data is deciphered. This critical step empowers the attacker to finally interpret and exploit the data for various nefarious purposes, such as identity theft or causing disruptions to business operations [16].

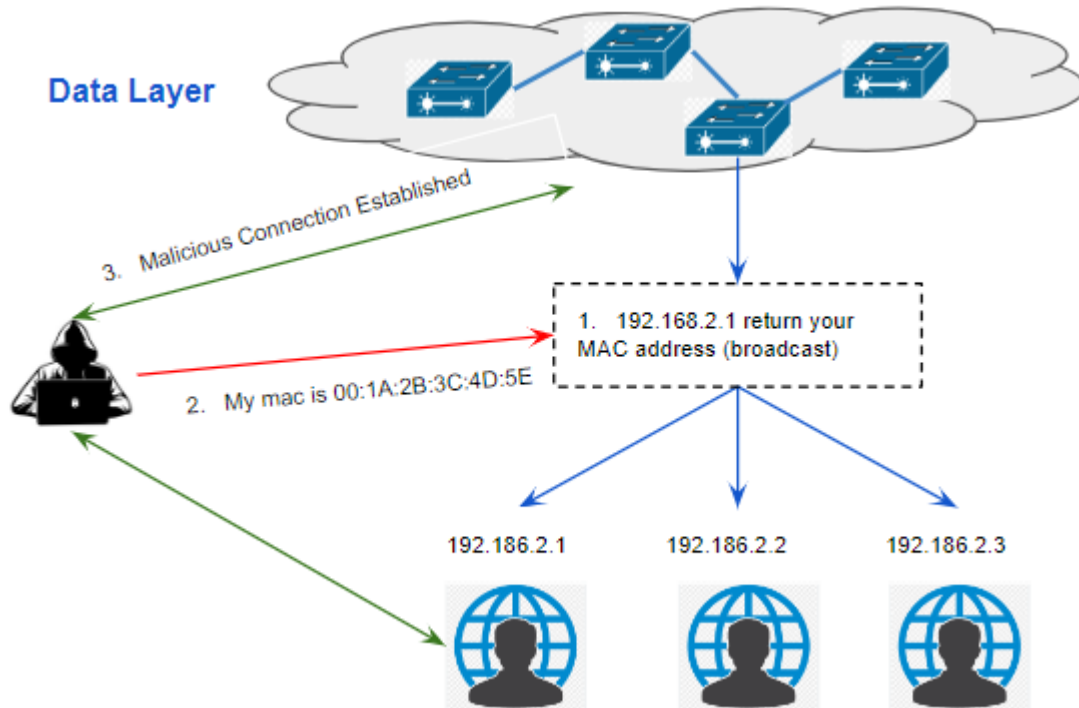


Figure 7, Depiction of man in the middle attack in SDN architecture

2.1.2 BGP Route Hijacking Attacks

BGP hijacking refers to a malicious activity wherein cyber attackers redirect Internet traffic by falsely claiming ownership of specific groups of IP addresses, known as IP prefixes, that they neither legitimately possess nor manage. This maneuver can be likened to an unauthorized alteration of road signs along a freeway, leading automobile traffic onto incorrect exits [17].

BGP, or Border Gateway Protocol, constitutes the backbone routing protocol of the Internet. It serves as the mechanism to efficiently direct traffic from one IP address to another. In essence, an IP address represents the actual web address of a particular website. When a user enters a website name, the browser relies on DNS (domain name system) servers to identify the corresponding IP address, while BGP offers the most efficient route to reach that IP address. In simplistic terms, if DNS serves as the Internet's address book, BGP functions as its road map [17].

Each BGP router maintains a routing table containing optimal routes between autonomous systems (ASes). ASes, which are often Internet service providers (ISPs),

continually broadcast updates concerning the IP prefixes under their control. BGP prioritizes the shortest and most direct path between ASes to reach IP addresses with the fewest network hops [17].

BGP's decentralized nature is instrumental in facilitating the expansive growth of the Internet. The Internet comprises multiple interconnected large networks, and the absence of a central authority dictating data packet routes necessitates the role played by BGP. Without BGP, web traffic could experience significant delays due to inefficient routing or even fail to reach its intended destination [17].

When an AS falsely announces routes to IP prefixes it does not legitimately govern, this announcement, if not adequately filtered, can propagate throughout BGP routers across the Internet. Consequently, traffic destined for those IPs will be unintentionally routed to that AS. In an analogous context, it is akin to asserting ownership over property without local government validation and enforcement [17].

BGP unfailingly favors the most specific and shortest route to the desired IP address. For a BGP hijack to succeed, the route announcement must either [18]:

- i. Propose a more specific route by declaring a narrower range of IP addresses compared to previously announced by other ASes (figure 8).
- ii. Offer a shorter route to specific IP address blocks. Furthermore, BGP route announcements must be made by AS operators or threat actors who have compromised an AS, albeit the latter occurrence is less common.

Despite the seeming audacity of large network or ISP operators engaging in such malevolent actions, the sheer global proliferation of over 80,000 autonomous systems underscores the potential for unscrupulous entities. Moreover, BGP hijacking is not always overt or easily detectable; malicious actors may obscure their activities behind other ASes or announce unused IP prefix blocks that are unlikely to raise suspicion, thus maintaining a low profile.

Consequences of BGP hijacking encompass misrouted Internet traffic, monitoring or interception of data, creation of "black holes" where data vanishes, and redirection to fraudulent websites as part of on-path attacks. Additionally, spammers may exploit BGP hijacking, or the networks of ASes involved in such activities, to spoof legitimate IPs for spamming endeavors. From a user standpoint, page load times can

increase due to inefficient network routing, possibly resulting in unnecessary global data traversal.

In the most benign scenario, traffic experiences increased latency due to the extended route. In the worst-case scenario, attackers may conduct on-path attacks or deceive users into visiting fraudulent websites to steal their credentials [18].

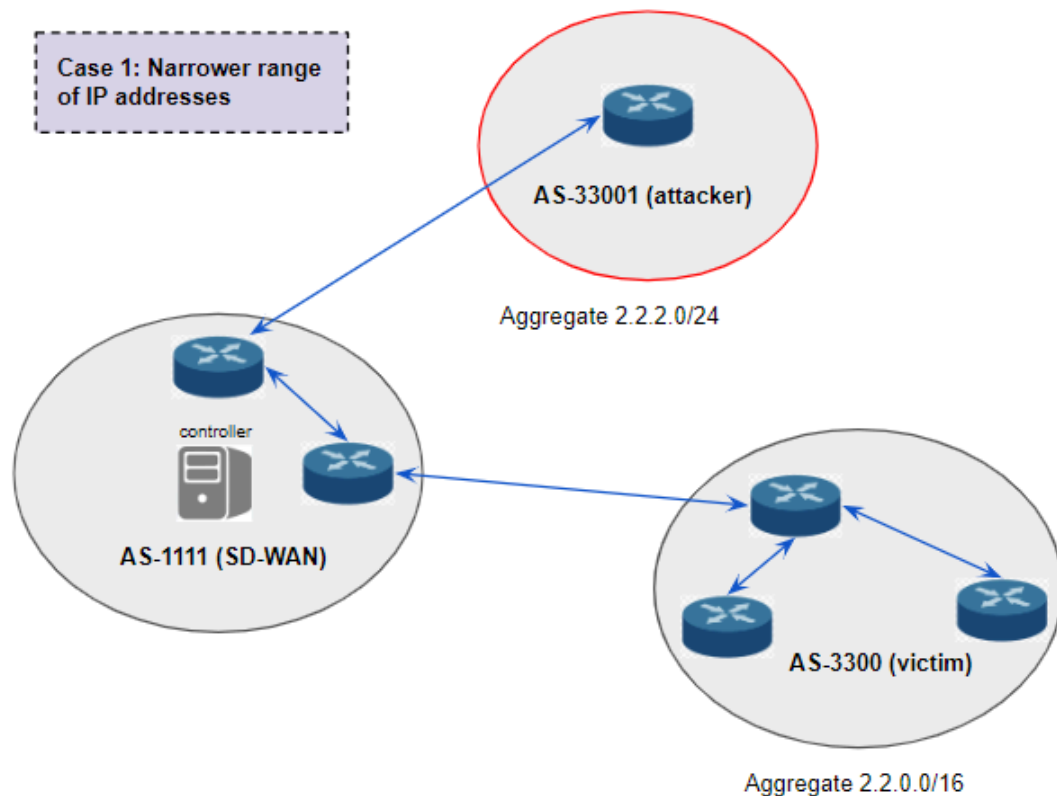


Figure 8, BGP hijacking by proposing narrower range of IP addresses

2.1.3 Sniffing and Eavesdropping Attacks

An eavesdropping attack, often referred to as a sniffing or snooping attack, entails the unauthorized acquisition of information during its transmission across a network via a computer, smartphone, or any other connected device. These attacks exploit insecure network communications, allowing malevolent actors to access data in transit as it is sent or received by the legitimate user [19].

The term "eavesdropping" may somewhat understate the severity of these incidents. Typically, attackers target sensitive financial and business data, which they can subsequently market for illicit purposes. There is also a burgeoning market for what is known as "spouseware," enabling individuals to eavesdrop on the digital activities of their loved ones, such as tracking smartphone usage [19].

Detecting an eavesdropping attack can prove challenging, as network transmissions tend to appear normal during the incursion. A successful eavesdropping attack hinges on exploiting a weakened connection between a client and a server, which the attacker reroutes for their benefit. The attacker deploys network monitoring software, colloquially termed a "sniffer," on a computer or server to intercept transmitted data [19].

Each device within the network, spanning from the transmitting device to the receiving device, constitutes a potential point of vulnerability (figure 9). Furthermore, the initial and terminal devices themselves may be compromised [19].

Eavesdropping attacks bear the potential for severe repercussions, including the loss of critical business data, the infringement of users' privacy, and the propagation of broader attacks like identity theft. A noteworthy illustration of the impact of eavesdropping attacks is their relevance to the growing use of digital assistants like Amazon Alexa and Google Home. While these assistants enhance users' convenience, they are susceptible to eavesdropping by attackers seeking to acquire sensitive information [19].

The consequences of eavesdropping attacks encompass [19]:

- 1) **Financial Loss:** Cyber attackers may exploit their access to sensitive information, such as corporate data, trade secrets, or user passwords, for financial gain. This could involve selling data to third parties or competitors or employing it as leverage in ransomware attacks, where access to data is blocked until a ransom is paid. Additionally, any instance of data compromise carries the risk of tarnishing an organization's reputation, potentially resulting in the loss of customers and, consequently, financial setbacks.
- 2) **Identity Theft:** Eavesdropping perpetrators can eavesdrop on conversations within seemingly secure applications, inadvertently causing users to divulge sensitive

information. This information can then be leveraged to steal user credentials and execute broader identity theft schemes.

- 3) Privacy Erosion: The pilfering of confidential information has the potential to infringe upon the privacy of businesses and individuals alike. Eavesdropping attackers can intercept critical business details, conversations, and exchanges, thereby compromising the privacy of affected users.

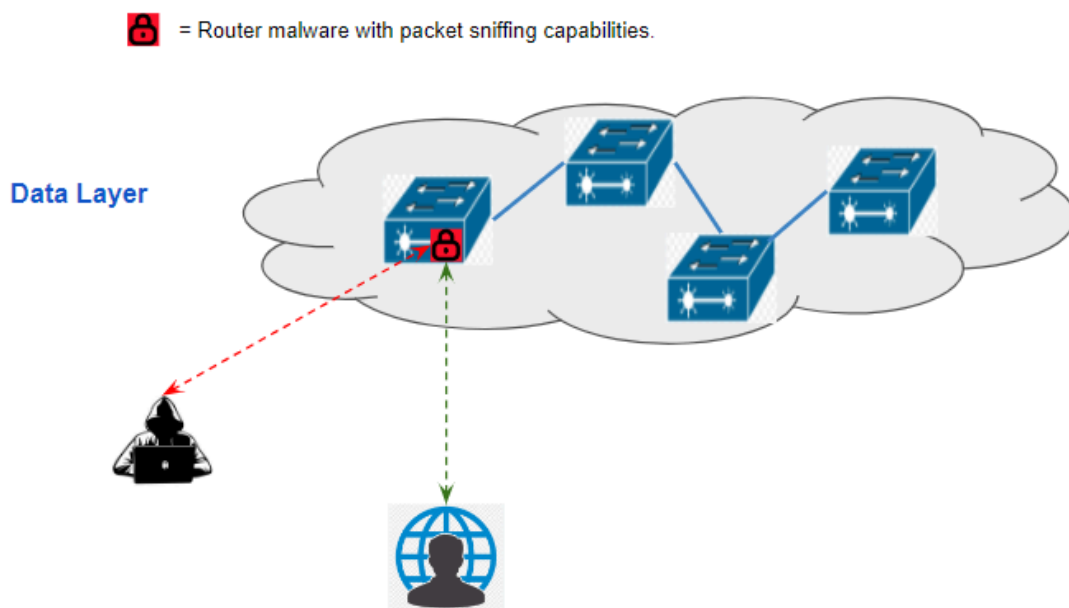


Figure 9, Eavesdropping attack via a vulnerable L2-L3 switch.

2.2 Attacks on the Application layer

2.2.1 Cross-Site Scripting (XSS) Attack

Cross-Site Scripting (XSS) attacks represent a class of security vulnerabilities wherein malicious scripts are inserted into trusted websites. These attacks occur when an adversary employs a web application to deliver malicious code, often in the form of client-side scripts, to another user. XSS exploits are pervasive and manifest wherever

web applications incorporate user input into their output without proper validation or encoding [20].

An attacker utilizes XSS to transmit malevolent scripts to an unsuspecting user. The victim's web browser, devoid of means to discern the script's untrustworthiness, executes it. As the script appears to originate from a trusted source, it gains access to any cookies, session tokens, or sensitive data stored within the browser, with the potential to rewrite the HTML page's content [20].

Cross-Site Scripting (XSS) attacks occur under the following circumstances [20]:

- I. Data enters a web application from an untrusted source, often via a web request.
- II. The data becomes part of dynamic content delivered to a web user without undergoing validation for malicious content.

The malevolent content, typically comprising JavaScript segments but potentially encompassing HTML, Flash, or other executable code, is conveyed to the web browser. This content may execute within the browser. The range of XSS attacks is extensive, but they frequently involve the exfiltration of private data (e.g., cookies or session information) to the attacker, redirection of the victim to content under the attacker's control, or the execution of malicious operations on the user's system, masquerading as actions performed on the vulnerable site [20].

XSS attacks fall into two primary categories: reflected and stored.

- 1) Reflected XSS Attacks: These attacks involve injecting scripts that are reflected off the web server, such as in error messages or search results. The victim typically encounters these attacks through alternate channels, like email messages or other websites. When a user interacts with a malicious link, submits a crafted form, or merely visits a malicious site, the injected code reaches the vulnerable web server, which then bounces it back to the user's browser. The browser executes the code, considering it originated from a trusted source. Reflected XSS is also known as Non-Persistent or Type-I XSS.
- 2) Stored XSS Attacks: In this scenario, the injected script is permanently stored on the target servers, e.g., in a database, message forum, or comment section. The victim retrieves the malicious script from the server when requesting the stored information. Stored XSS is also referred to as Persistent or Type-II XSS (figure 9)

- 3) Blind Cross-site Scripting: This type is a form of persistent XSS where the attacker's payload is saved on the server and reflected back to the victim through the backend application. It may be challenging to confirm in real-world scenarios, but tools like XSS Hunter can assist in this context.

Apart from Stored and Reflected XSS, there's another type known as DOM-Based XSS, identified by Amit Klein in 2005. It's organized into a matrix of Stored vs. Reflected XSS and Server vs. Client XSS, with DOM-Based XSS as a subset of Client XSS, as recommended by OWASP [20].

XSS attacks can yield severe consequences, irrespective of whether they are stored or reflected. The distinction lies in how the payload reaches the server. Notably, even seemingly read-only or brochureware sites can be susceptible to significant reflected XSS attacks. These attacks can lead to a range of issues, from minor annoyances to complete account compromise. The most serious consequences include the exposure of a user's session cookie, enabling an attacker to hijack the user's session. Other potential impacts encompass file disclosure, installation of Trojan horse programs, redirection to malicious pages, or content manipulation. For instance, an XSS vulnerability on a pharmaceutical site could result in erroneous dosage information and potential overdoses, while an attack on a corporate website could influence stock prices or erode consumer trust [20].

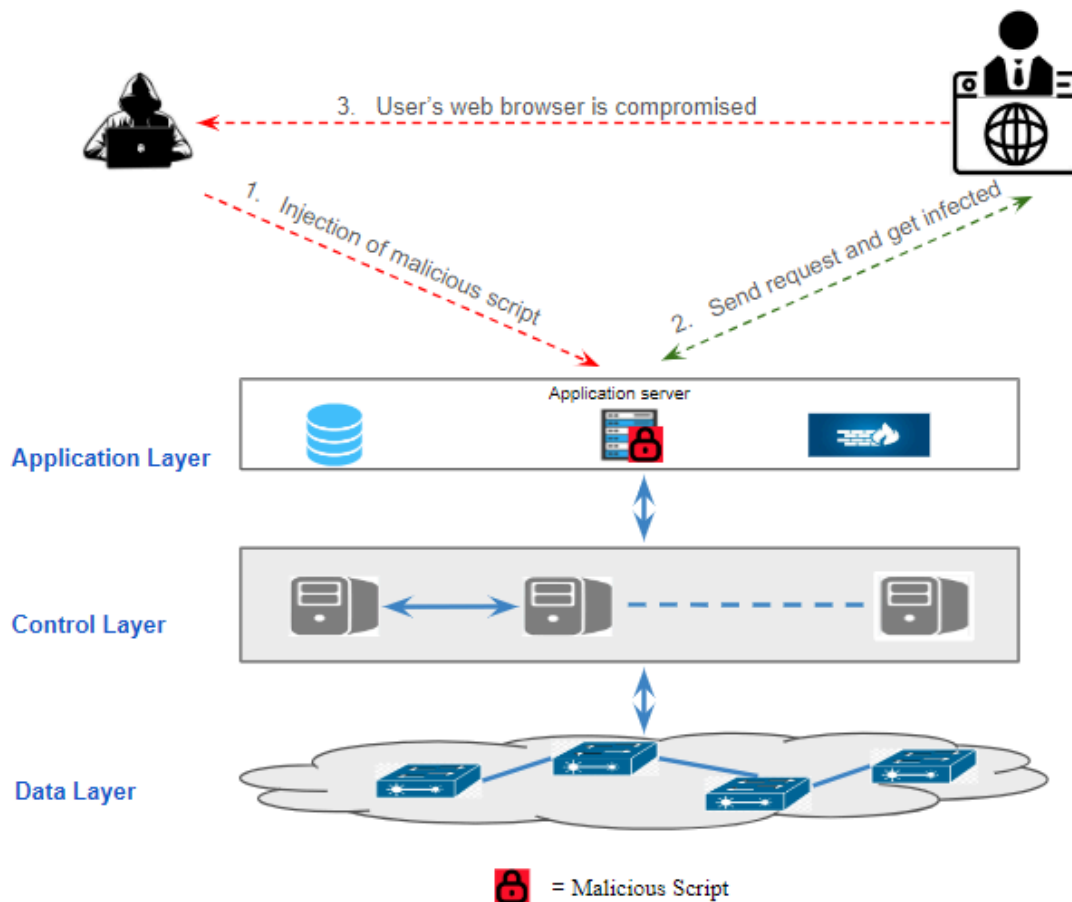


Figure 9, Depiction of a persistent type two XSS

2.2.2 SQL Injection Attack

A SQL injection attack entails the insertion or "injection" of a SQL query through input data from the client to an application. A successful SQL injection exploit can have severe repercussions, including reading sensitive database data, modifying database records (Insert/Update/Delete), executing administrative operations on the database (such as shutting down the DBMS), retrieving content from specific files within the DBMS file system, and, in certain cases, issuing commands to the underlying operating system. SQL injection attacks fall under the category of injection attacks, where SQL commands are injected into data-plane inputs to manipulate the execution of predefined SQL commands [21].

SQL injection attacks empower malicious actors to impersonate others, manipulate existing data, create repudiation issues (e.g., voiding transactions or altering balances), disclose all data present in the system, destroy data, render it inaccessible, or even gain administrative control over the database server [21].

These attacks are notably common in PHP and ASP applications due to the prevalence of older functional interfaces. Conversely, J2EE and ASP.NET applications are less susceptible to easily exploitable SQL injections owing to their nature and the availability of programmatic interfaces [21].

The gravity of SQL injection attacks is contingent on the attacker's proficiency and creativity, as well as the effectiveness of defense mechanisms, such as employing low-privilege connections to the database server. In general, SQL Injection is deemed a high-impact vulnerability [21].

A SQL injection attack occurs when [21]:

- 1) Unintended data infiltrates a program from an untrusted source.
- 2) This data is used to dynamically construct a SQL query.

The primary consequences encompass:

- i. Confidentiality: SQL databases typically store sensitive data, and SQL Injection vulnerabilities frequently lead to the loss of confidentiality (figure 10).
- ii. Authentication: Poorly constructed SQL commands for user authentication can potentially enable unauthorized access to a system, allowing an attacker to log in as another user without prior knowledge of their password.
- iii. Authorization: If authorization details are stored within a SQL database, successful exploitation of a SQL Injection vulnerability can permit the alteration of this information, potentially granting unauthorized access.
- iv. Integrity: In addition to the potential for reading sensitive data, SQL Injection attacks can facilitate data modifications or deletions, jeopardizing data integrity.

SQL Injection attacks pertain to the SQL language and can affect any platform that interacts with a SQL database. These attacks are notably prevalent in database-driven websites. Detecting this flaw is relatively straightforward, and its exploitation is uncomplicated. Consequently, any site or software application, even those with a

modest user base, is susceptible to attempted SQL injection attacks. The attack is essentially executed by introducing a meta character into data input, enabling the injection of SQL commands into the control plane, where they were not intended to exist. This vulnerability exploits the absence of a clear distinction between the control and data planes in SQL [21].

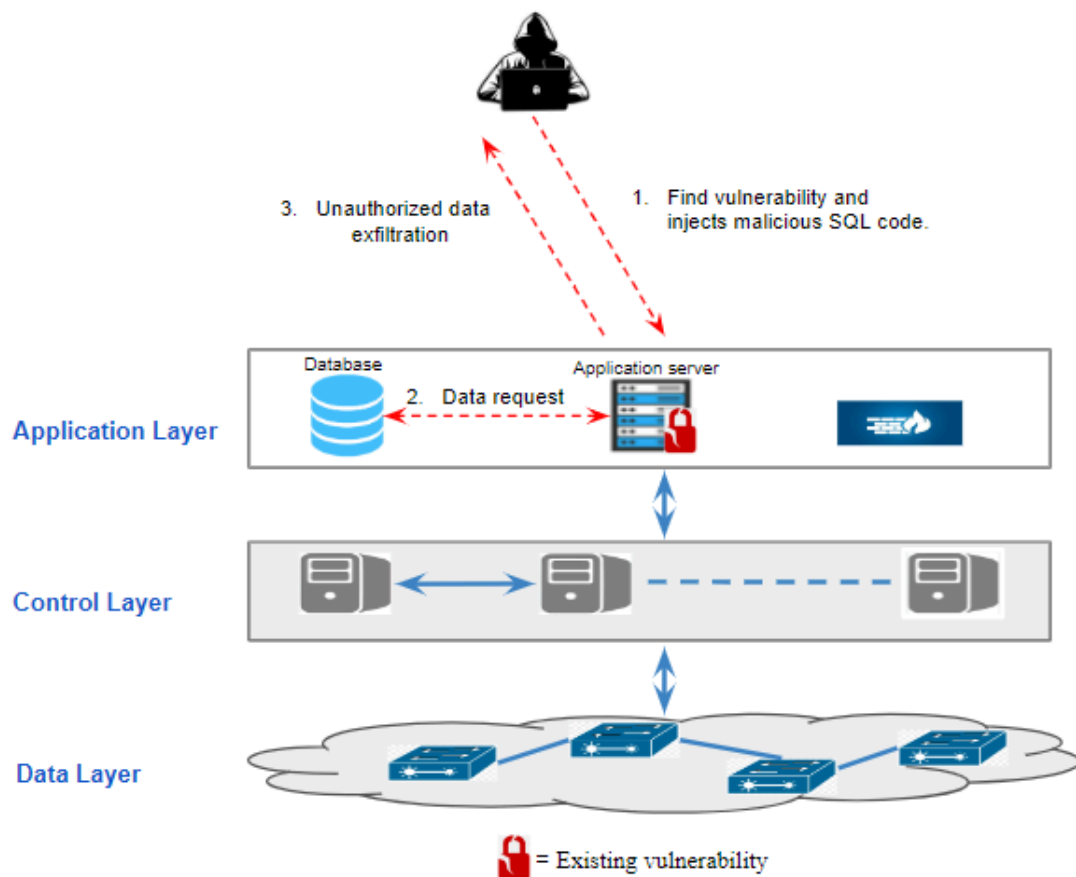


Figure 10, SQL injection on vulnerable application server

2.2.3 Cross-Site Request Forgery (CSRF) Attack

Cross-Site Request Forgery (CSRF) represents an attack vector in which an adversary compels an authenticated user to unwittingly execute unauthorized actions on a web application. Employing social engineering techniques, such as sending deceptive links via email or chat, the attacker manipulates users into performing actions specified by the attacker's intent. If the targeted user is an ordinary account holder, a successful CSRF attack can coerce them into executing actions that alter the application's state,

such as fund transfers or changes to personal details. In the case of an administrative account compromise, CSRF can jeopardize the entire web application's security [22].

CSRF operates by inducing the victim to submit a malicious request, effectively inheriting the victim's identity and privileges to carry out undesired functions on their behalf. Notably, most web requests automatically include credentials associated with the site, such as session cookies, IP addresses, and Windows domain credentials. Consequently, when a user is authenticated on the site, the site cannot differentiate between the falsified request initiated by the victim and a legitimate request from the same user [22].

These attacks target functionalities that lead to alterations in the server's state, encompassing actions like modifying email addresses, passwords, or conducting transactions. CSRF is particularly potent in gathering the victim's private data through a variation known as login CSRF, where the attacker coerces a non-authenticated user to log into an account controlled by the attacker. Subsequently, the attacker gains access to the victim's added personal information, such as credit card data [22].

Stored CSRF vulnerabilities, also known as "stored CSRF flaws," may occur when the attack is saved on the vulnerable site itself. This can be accomplished by embedding an IMG or IFRAME tag in a field accepting HTML or through more intricate cross-site scripting methods. The severity of such an attack is amplified as it increases the likelihood of victim interaction, as the victim is more likely to access the page containing the attack while already authenticated to the site [22].

CSRF attacks are referred to by various names, including XSRF, "Sea Surf," Session Riding, Cross-Site Reference Forgery, and Hostile Linking. Microsoft terms this type of attack a One-Click attack in their threat modeling process and documentation [22].

To execute a CSRF attack successfully, attackers must grasp how to generate a valid malicious request for the victim to unwittingly enact. For instance, in the context of a vulnerable bank.com web application, an attacker named Maria aims to deceive Alice into transferring money to her account. Maria follows these steps [22]:

- 1) Constructs an exploit URL or script.
- 2) Tricks Alice into executing the action through social engineering techniques.

This involves generating an exploit URL (figure 11) that manipulates the request to transfer a substantial amount of money from Alice's account to Maria's. The social engineering aspect involves inducing Alice to access this URL while logged into the bank application, often achieved through methods such as unsolicited emails with HTML content or planting exploit URLs on pages frequently visited by the victim during online banking [22].

The disguised exploit URL can take the form of an ordinary link or a concealed image tag, urging the victim to click it. In either case, the browser submits the request to bank.com without visual indication to Alice that the transfer has transpired [22].

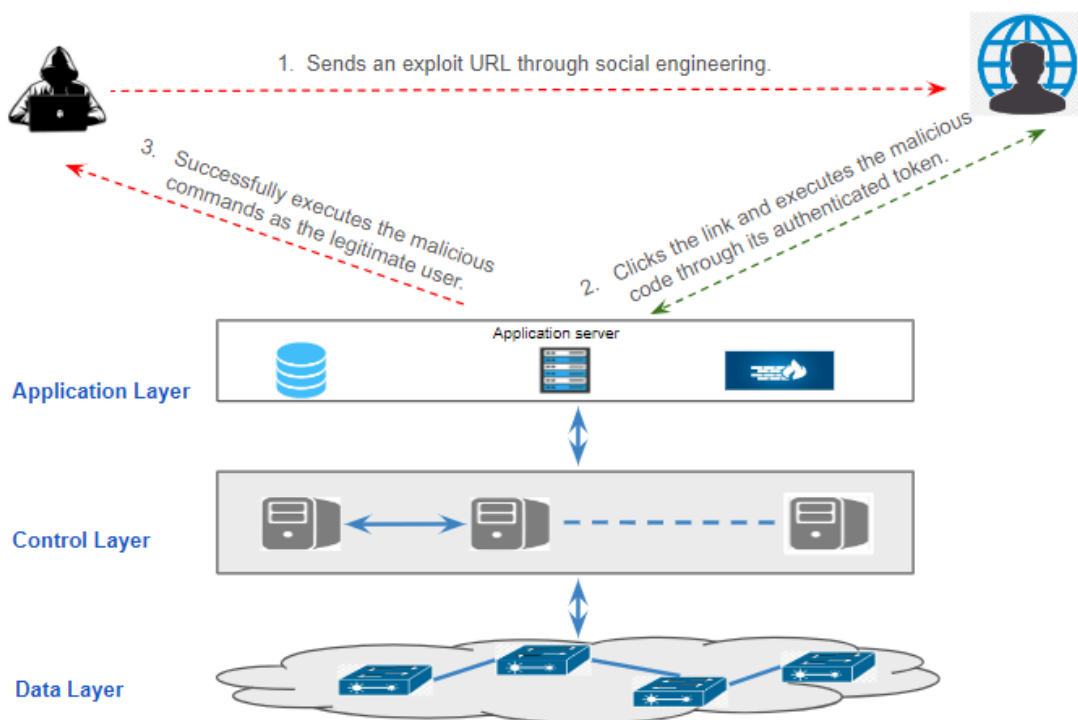


Figure 11, Deploying CSRF attack through social engineering techniques

2.3 Attacks on the Control layer

2.3.1 DDOS attack to controller by exploiting session hijacking:

One of the most classical attacks is host location/session hijacking[24]. The attacker steals the session between the user and the Web Server and through the session

succeeds in invading the first defensive line (firewall). Given the type of controller the host profile is being authenticated and authorized, based on some elements such as the location, IP, MAC of the remote user.

However the attacker, after stealing the session, can impersonate the location, IP and MAC by altering the packet in order to deceive the security checks. This type of attack can be used for three different purposes.

1) The first purpose is to launch a DDOS attack against the control/data layer and threaten the controller's availability. We should keep in mind that if the system is supported by a unique controller then it constitutes a single point of failure (figure 12).

2) The second purpose is to grant access in the web server and from there to exploit any possible vulnerability.

3) The third and most classical case is to steal sensitive personal data.

The most common defense tool for these kinds of attacks is the implementation of a Multi-layer IDPS (intrusion and detection protection systems). IDPS will be analyzed further in the Cyber Defense solutions of SDN. Furthermore, it should be pointed out that the security level against session hijacking of each SDN ecosystem depends on the controller's technology and properties.

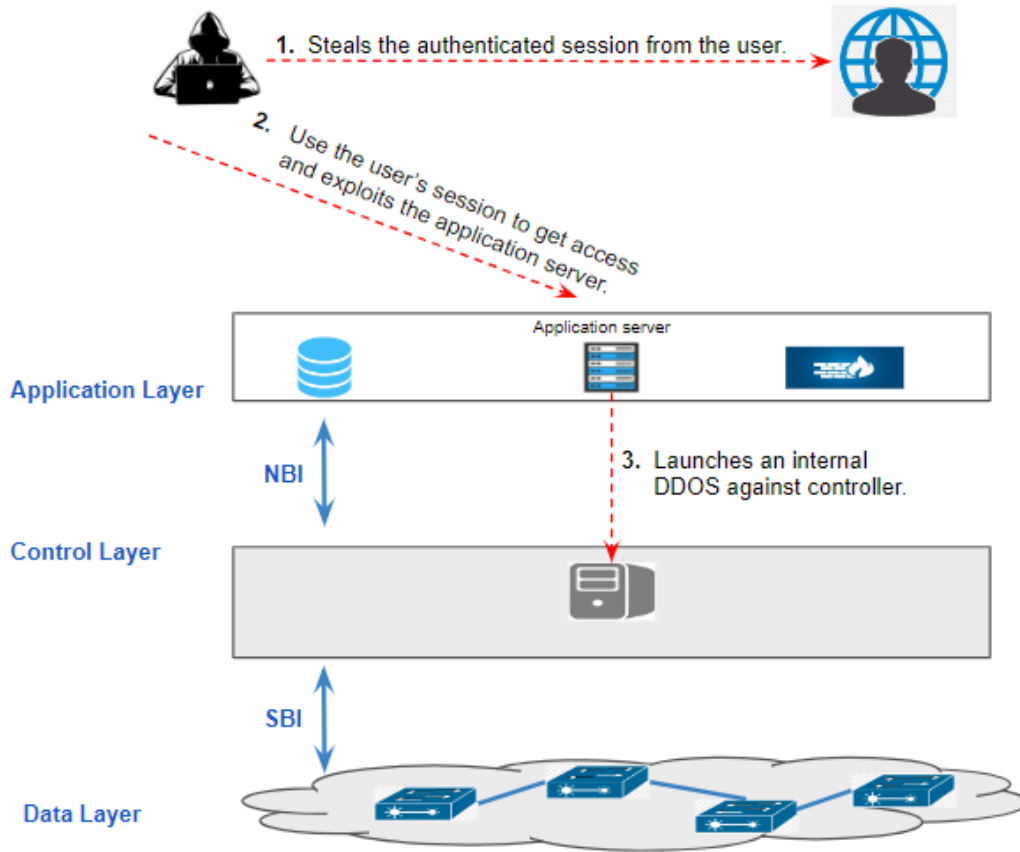


Figure 12, Launching internal DDOS by exploiting session hijacking

2.3.2 Link fabrication attack

Description

An attacker who tries to launch an LFA attack will try to modify the current controller's mapping of the network topology. Through this alteration will pass a total new link which can be exploited 100% percent by the attacker. The majority of SDN systems(Floodlight, OpenDayLight, etc.) discover the topology of the L2-L3 switches from LLDP frames. The attacker through known communication protocols such as LLDP(link layer discovery protocol) will try to add the malicious link in the topology.

Simulation

This attack exploits the LLDP frames sent between the network appliances. There are 3 common categories of attacks, each of them is based on generate, replay or relay LLDP frames respectively. The first two categories, generation and replaying of LLDP can be prevented by using in the authentication a one time token. In the third category of attacks still there is not , the attacker must gain vision between switches flow in order to watch the relay of the LLDP frames[25]. After gaining visibility, the attacker will try to place a malicious host between two network appliances in order to bridge the two interfaces. The malicious author will analyze the current topology of the network and choose 2 switches in different networks so as to increase the possibility of confusing the controller. The next step is to make the new link “interesting” for the SDN by offering a low hop(faster path than the other adjacent switches) so as to add the fabrication link to the current topology.

Defensive technique against LFA

The defensive solution is based on a statistical analysis of the latencies of each new link (figure 12) [25]. The two extra hops (at least) of the fabricated link will increase the latency for the relayed traffic, thus the controller can calculate the variations in network link latencies and decide If the new link is fabricated or not. Actually the controller during the initialization of the network keeps a baseline of the latencies of known benign links. As a consequence, when a new link in the network is advertised to the controller, the controller before making the new link a new node in its premises, passes the link through a vetting period. The vetting period starts by sending many frames until gathering a quite adequate sample of latencies from the new link. Then the controller compares the sample of the new link with the baseline and If the possibility of the new link to be fabricated is bigger than a threshold, the controller rejects the potential link and raises an alarm.

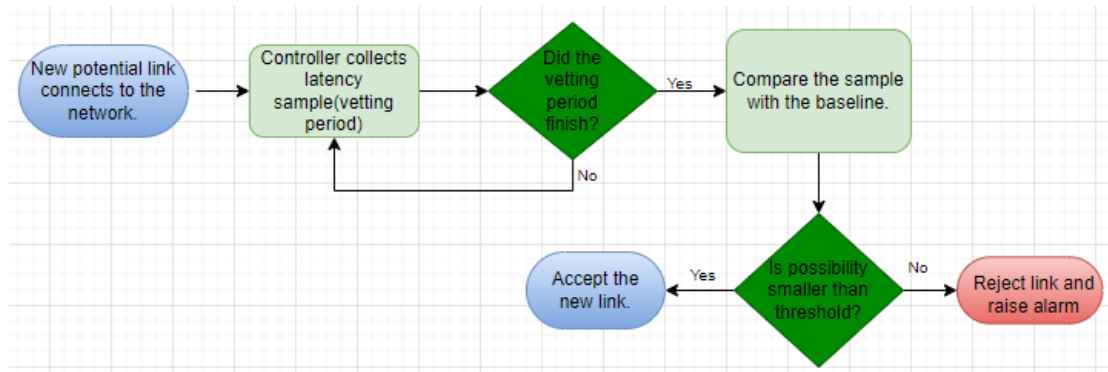


Figure 12, Flowchart of LFA defensive technique

2.3.3 Flow table overloading

One of the most common attacks both in conventional networks and SDN is the flow table overloading [26]. This method is implemented by the attacker by sending a huge number of packets from new-unknown IPs to the Data layer (figure 13). This attack affects both the data and control layer and it can cause delays and even affect the availability of the controller and the L2-L3 switches.

- 1) Resource Exhaustion: The switches may have limited resources allocated for storing flow entries. When the flood of flow modification requests surpasses the available resources, the switches may become overwhelmed and start dropping the legitimate traffic.
- 2) DDOS against controller: When a switch receives a packet it tries to parse the flow table and match it with an existing destination IP address. If the IP is not included within the flow table, then the switch generates a "Packet-In" event (sending the whole packet) to the controller so as to decide what to do with the specific packet.

The SDN controller examines the packet and makes a decision on how the packet should be handled. This decision may involve installing new flow entries in the switches, updating policies, or taking other actions such as discarding the packet and blocking this source IP. If the number of packets is quite big it can cause a dangerous DDOS with tremendous consequences especially for the controller.

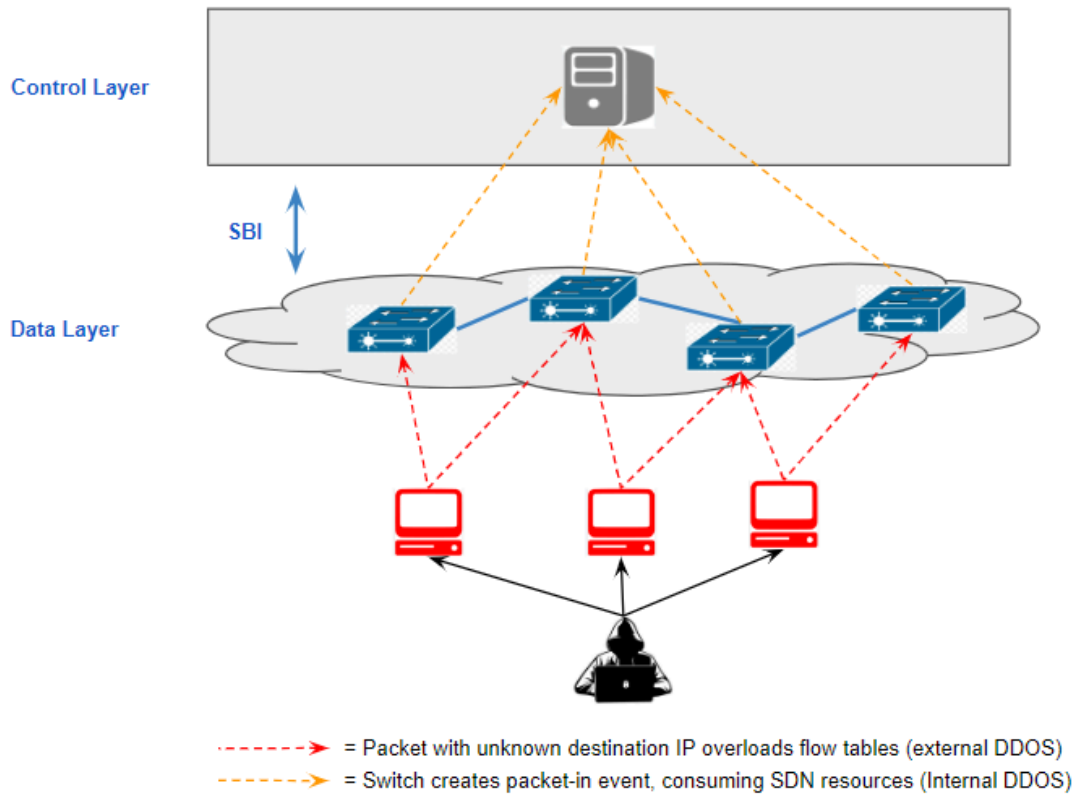


Figure 13, Launching internal and external DDOS through a botnet.

Table: Classification of all security attacks per SDN plane.

SDN Layer	Attack Type	Impact of attack
Application	Cross-Site Scripting (XSS) Attack	Steal session cookie and inject malicious code in user's browser.
	SQL Injection Attack	Unauthorized access to confidential data and reputation damage.
	Cross-Site Request Forgery (CSRF) Attack	Executing remotely unauthorized commands.
Control	DDOS attack to controller by exploiting session hijacking	Bring down the controller and cause long latency.
	Link fabrication attack	Creating a black hole, exfiltrating data or launching any attack.
	Flow table overloading	Damage availability of all network devices

		and controllers.
Data	Man-in-the-middle attack	Data interception Identity theft.
	BGP Route Hijacking Attacks	Cyber espionage and threaten global internet stability.
	Sniffing and Eavesdropping Attacks	Wide privacy violation and launching reconnaissance phase.

Chapter 3: Holistic defense strategy of SDN deployed in 5

basic axes

3.1 Policies in SDN:

Policies in SDN are one of the most important pillars in order to handle packet routes, assure the QOS(quality of service) and last but not least to protect the security of networks from potential attacks. Policy definition contains the concept of traffic identifiers. There are 2 different categories of traffic identifiers, the low level traffic identifiers (LLTI) and the high level traffic identifiers (HLTI) [27].

The LLTI are IP addresses and port numbers and are included in each packet to serve fundamental connectivity purposes such as routing and switching. The HLTI are the user name and application name and as we will analyze below, helps to create more clear, robust and explicit policies.

The High level SDN model is used in Application Aware networks. As we know nowadays data centers follow and are aligned with an application driven path and their architecture is designed on this aspect.

It's unambiguous that any simple SDN controller can not be promoted to a High Level SDN. If there are no server applications connected to the underlying layer, there is no traffic metadata. Thus the plain SDN model usually employs only LLTI to create policies.

The total number of SDN HLTI is called a High Level SDN. The high level SDN has the advantage to categorize the flows deriving from the same user, same application or machine name. This categorization gives many capabilities to the network engineer and allows to make massive modifications in an easy, prompt and robust way.

However there is a prerequisite so as to work the High Level SDN. Traffic metadata is absolutely necessary to create all the policies. The High Level SDN gathers all the

traffic metadata and pushes them to the application plane via the northbound API and creates new rules or modifies the existing ones based on HLTl results.

Furthermore, High level SDN helps to reduce the number of rules in policies, eliminate the overlaps (and all the negative effects of overlapping) and in the end make as simple as possible all network policies.

How high level SDN collects traffic metadata and exploits them

High level SDN collects high level information mainly from 3 different sources[27]. The first source is the application managers servers, the second source is the enhanced host(are connected with applications) and the third source is monitoring tools such as a Deep Packet Inspection(DPI) engine.

After the collecting procedure (figure 14), the first phase is the evaluation of the collected data. Usually the evaluation of the data is connected with the reliability of the sender/source. A controller ranks its sources with weight factors based on the target. The most common tactic is to give the bigger weight to the server managers application and the less weight to enhanced users[26]. After evaluating the HLTl, the high level SDN stores and analyzes the metadata given the existing policies(created by the administrator in the application plane) and modifies the data plane according to the needs. To summarize the above procedure, the HLTl feeds the models of existing policies with the purpose of reducing the complexity of the network flows and increasing the efficiency of networks. The last step is to create a relationship/mapping from high level traffic identifiers to low level traffic identifiers in order to push the new modifications in network flows from controller to data plane. Finally it's clear that the High Level controller by analyzing the traffic metadata instantly increases the consistency and the security of the flows for the whole network.

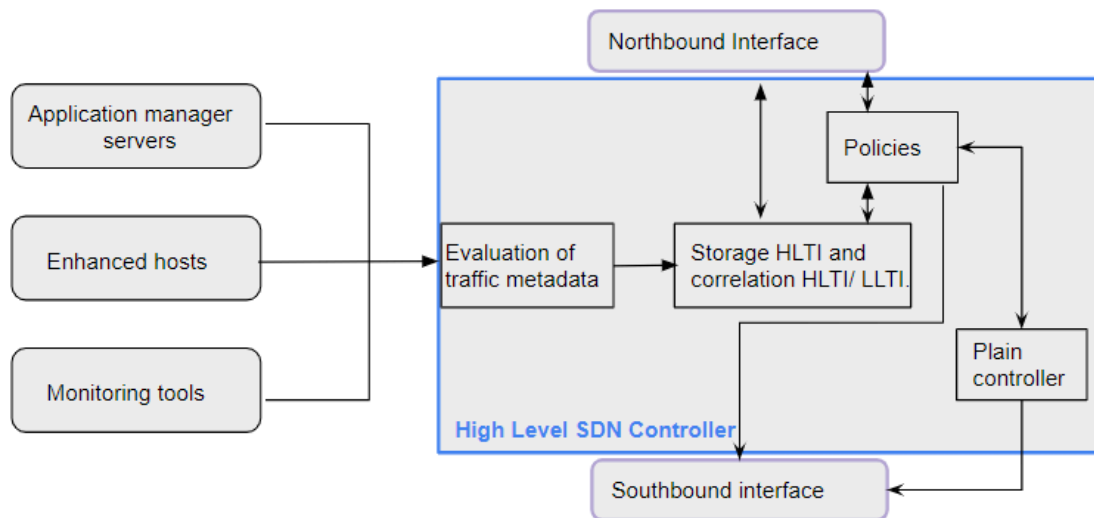


Figure 14, Depiction of High Level SDN Controller

Now we can understand the importance of metadata control and the consequences of injecting false metadata into a High Level controller. Thus the communication between metadata sources and the High level controller should be protected from multi-factor authentication.

Two different models of High Level SDN:

As it has already been mentioned, many technologies such as data centers have application centric architecture. Application Aware Networks (AAN) is a model of High level SDN architecture and it was developed to serve and improve the network mapping of applications [28]. An AAN model needs many HLT/ to work properly as application name(unique identifier), Device type(phone, computer, etc) and media type(type of the data). For example, for sending images through a specific application choose a link with low bandwidth.

The second model is the Identity Aware Networks(IAN) and it's used for developing SDN for businesses or generally medium size networks. The information - HLT/ that needs to work is based on the people and more specifically the Person ID and the Group that the person belongs to. For example, give priority to all packets that are connected with the Group of the management team.

AAN and IAN can work quite efficiently and provide great services and access control. If their utilization is combined. For example, by defining specific policies we can give access to the payroll application only for the human resources department.

3.2 Distributed firewalls and microsegmentation

The conventional firewalls can not support and optimize either the SDN or any multi-tenant virtualized environments. Thus, in order to support the rapid progress of SDDC (software-defined data centers) it was developed the distributed firewall (DF). The distributed firewall is deployed in many different places inside the architecture and it is centralized and managed by the controller [35]. The three basic components of the DF are the security policies, the orchestrator of the policies and of course a protocol to assure that the communication between the controller and the firewall nodes is secure. The orchestrator keeps records of all the correlation between all the policies and the SDN and usually the communication protocol is IPsec. It should be pointed out that nowadays the host identity is based on cryptographic certificates rather than IP because the former is considered much more secure. and adds a new significant principle, the topology independence and this condition increases the degree of deploying freedom. In comparison with the conventional solutions which can not support and scale with SDN/NFV, the DF provides many primary advantages like [26]:

- Internal protection: It's crystal clear that in large network scales and in data centers the internal attacks are a huge issue, which can be faced only with firewalls in many different topology spots. In data centers it can not be deployed multi-tenant services without first deploying a distributed firewall. Thus DF increases general the security of internal and external flows.
- Defensive against APT: It is known that even the largest organizations are vulnerable against some Advanced Persistent Threats. Distributed firewalls can disrupt many different phases of the lifecycle of an APT, such as the internal reconnaissance and the lateral movement. The majority of APTs will

pass through the lateral movement phase in order to maximize their damage/consequences. Furthermore, when a VM is compromised it's much easier to mitigate it with the purpose of reducing the total infection of the ecosystem (works well against Malware propagation).

- Topology independence: As it is written above, by using certificates the controller can recognize all the nodes of distributed firewalls even if these are not placed in a specific perimeter of the network. Also VXLAN technology can be utilized through the above independence because we can provide L2 segments with L3 isolation. Furthermore, it is much easier to migrate a whole VM in a secure way and to allow the communication between different nodes which are located in different Data Centers.
- Upgraded performance and confrontation of SPOF(single point of failure): Given the nature of DF, all the nodes can work simultaneously and thus the whole workload of packets is divided, so the performance is increased. Moreover through DF we can utilize the VXLAN protocol. In parallel while the performance is increased the possibility of happening a SPOF incident is decreased and in the end even a SPOF incident happens will affect a much more narrow area. If we try to install physical firewalls then the latency will be increased exponentially because of the triangular geometry that the flows will be driven through.

The deployment of a distributed firewall[29] creates a microsegmentation of the initial environment, which increases the security posture of data centers. The microsegmentation allows us to define policies and manage all the known defensive solutions (IDPS, DPI, etc) in the endpoint layer. Nowadays, the two most common solutions providing microsegmentation are the ACI (Application Centric Infrastructure from Cisco) and the NSX of VMware which is a standalone platform providing logical networking and security capabilities and services, such as switching, routing, firewalling, load balancing, virtual private network (VPN), quality of service (QoS), and monitoring.

NSX and ACI technologies beyond the DF(distributed firewall) also provide the ESG(Edge Services Gateway). The ESG is located and operates at the network edges to secure the perimeter of the virtualized perimeter, thus it is called gateway. ESG monitors the traffic between the physical to virtual machine primarily at layer 3.

Microsegmentation utilizes the creation of zero trust zones by deploying in each segment policies and rules,increases the granularity and the efficiency of the defensive tools and the management is conducted through a centralized plane. Last but not least microsegmentation provides an application firewall regardless of the underlying L2/L3 network topology. While all the infrastructure services pass from the network appliances to the software functions, it becomes possible to develop an application centric(topology independence) segmentation. Through this procedure, the underlying network layers do not affect the routing/security mechanisms. In this case, all the flows are based on the L4-L7 layer and pass through multiple functions(also called chain services) [26]. Usually there are two basic categories of chain services: the network services and the guest services. The network services usually serve the data in transit and inspect all the low level traffic identifiers while the guest services are applied on the endpoints (data on rest) and primarily concentrate on computing and storage units. It is clear that the above services/functions may be very demanding regarding the resources that need to work. Thus are developed the SVMs (Service Virtual Machines) which receive the data directly through a communication channel with the hypervisor. The biggest benefit derives from microsegmentation is the capability to provide efficient security services independently from the underlying topology(physical or logical). Furthermore the service functions are strictly correlated with the policies and vendor independent. Also it should be noted that each VM can have one or many security tags in order to be recognizable. The security tags are useful to classify the VMs in security groups. The attributes of the security tags are numerous such as the department, the vulnerabilities, the OS, even a data type classification(CRM services).

The latter allows us to change the conventional way of security services and by dividing the payload from the network attributes to increase the performance of the data centers.

The service composer constitutes the basic tool which enables the security services in modern data centers. The service composer can be divided into 3 different parts:

- Intelligent grouping: This concept refers to the separation between the workload and the network topology by using the security groups. A security group is a mechanism which collects both static(all the information regarding the infrastructure such as Active directory group, MAC, etc) and dynamic (all the information regarding the VM as the operating system, the security tag,computer name, etc)conditions and desires of the administrator and create a group with specific prerequisites. The dynamic attributes dominate over the static ones and are more flexible. The controllers always evaluate the above procedure.
- Service recording and deployment: It's a process/function which allows the third-party vendors(other companies or organizations) to record and deploy their services, applications or solutions into a data center environment. Sometimes also involve general details, its capabilities, its requirements and other information the data center administrators need so as to evaluate and approve the service.
- Security policies: Security policies which are provided by default by the platform or there is additionally the opportunity to be modified (custom tailored software) by the tenant in order to optimize the performance and the security of the VMs.

Finally the firewalls leverage the security groups and create a defensive line more security, more agile, more adaptive and generally speaking it is developed a more workload-centric approach which covers and suits better to the needs of a modern large, rapid scale data center.

NIST understood the high importance security of virtualized environments and the necessity of giving clear instructions on how to apply the firewalls [26] :

- While the VM hosts an application with high input and output flows then the virtual firewall should be run on the kernel of the hypervisor in order to take advantage of the hardware specifications and increase the speed and total performance. Thus, the latter implementation is called kernel-based virtual firewall.
- The second instruction is that both kernel-based or subnet-level should be integrated with a virtualized platform so as to increase the management of the workload, the visibility and it will reduce the configuration errors.
- Last but not least, for both types of firewalls it is preferable to work with high level rules, high level policies/groups and high level traffic identifiers (HLTI). This countermeasure utilizes the microsegmentation and its granularity immediately.

3.3 Multi-layer Intrusion detection and prevention Systems(IDPS):

As it is known one of the most significant defensive tools for each network is the IDPS. Even in the conventional networks IDPS is placed between the firewall and the routing network appliance and dramatically increases the overall security. IDPS is also being deployed in SDN but due to the different architecture there are many alternative solutions, but let's dive in.

The first categorization can of IDPS is based on where it is deployed. The 2 basic categories are based on the deployment entity.

The host-based IDS(HIDS) is deployed on a specific endpoint and is designed to protect against internal and external threats. Such a system has the ability to monitor network traffic to and from the endpoint, monitor processes running on the local machine, and inspect system logs. So a HIDS has very good visibility into that particular machine only but falls far short of network monitoring.

The network-based IDS(NIDS) is a solution to monitor the entire network. It has visibility into all traffic flowing through the network and makes decisions based on packet content and metadata. This approach provides more information and has the ability to detect broader threats. However, NIDS does not provide us with information about the internal endpoints.

Apart from where the IDS is deployed (network or computer), IDS are divided based on how they identify threats into 3 main categories: signature detection, anomaly based detection) and the hybrid detection system (Hybrid detection).

Signature detection: IDS that search based on signature essentially use the fingerprint of already known threats. So there is the threat fingerprint stored in a database and compared for identification. When a new threat is detected the system database is automatically updated. This detection method is very effective against already known threats (with a known fingerprint) and the false positive alarm rate is almost zero. However, it fails to detect new threats and vulnerabilities and is blind to zero-day vulnerabilities.

Anomaly based detection: IDS of this type essentially try to build a descriptive model of the behavior of the average user of a system. It then compares the various behaviors/movements in relation to the model that has been created and when there is a difference the IDS considers the movement malicious. This particular method is quite capable of finding new attacks or zero-day attacks, but it is very difficult for someone to make a correct descriptive model, and the consequence of this is false alarms and the non-detection of real threats. As we will analyze in the next section, artificial intelligence and its subcategories plays a dominant role in these systems.

Hybrid detection: IDS of this type is based on both the footprint of already known threats and the behavioral model. It is often the best solution to defend in the modern world.

The majority of existing SDN solutions have adopted a hybrid approach IDPS regarding the identification of threats. However there are many different

models/strategies on where to deploy the IDPS given that the infrastructure in SDN is elaborated.

The exact location and deploying strategy of IDPS within the SDN varies given the existing conditions such as the network size, the utility of the SDN(WAN, LAN, DATA CENTER), the traffic volume, the performance of requirements and the security objectives given the criticality[24]. The chosen strategy should allow effective monitoring and analysis, immediate response to security incidents while considering the scalability, the visibility and the coordination of the SDN. Another significant factor is the topology of firewalls within the environment. Usually the IDPS are located exactly after the firewalls.

- **Centralized IDPS:** In some architectures the IDPS is integrated within the controller and it is strictly correlated with an application firewall also deployed in the controller. This allows the controller to have a better monitor, analysis and supervision of the whole entity of network traffic and appliances. If there is a cluster of controllers, then an IDPS is deployed in each of them and almost always the IDPS communicate with each other in order to provide better results and visibility of all domains.
- **Distributed IDPS:** Another strategy is to find the most focal points of ingress and egress data volume and deploy over there the NIDPS. Some key points in SDN are the edge switches(L2/L3). Moreover, a distributed model can be deployed in all network appliances and be coordinated by a central server in the application plane or in the controller.
- **NFV IDPS:** Another common tactic is to deploy a host-based IDPS behind the main firewall in the virtualized application plane behind the virtual firewall. From the one hand the flows will pass through the whole network to reach the application plane, but on the other hand this IDPS will have increased efficiency, because the packets will be unencrypted. Thus the IDPS can perform Deep Packet Inspection(DPI).

Most of the time, the deployment of IDPS[26] takes place in many locations inside an SDN architecture. Usually sensors are installed before the packets pass through the Openflow protocol from the network appliances and in parallel inside the controller there is a host based IDPS. Each of these IDPS is focused and runs a unique analysis. For example the first IDPS it's more possible to have only a signature detection capability enriched with CTI(cyber threat intelligence) elements such as low level traffic identifiers (LLTI) so as not to create latency, while the one in the controller has more rounded analysis before entering the application plane where the DPI will be applied. By following this schema we have a Multilayer IDPS meaning that all packets will pass through a holistic analysis of signature and behavioral based.

If the IDPS is deployed in the application plane, after the firewall, many attacks can happen. For example If a massive DDOS happens, the data and the control layer will become 2 points of pressure until the IDPS understand the DDOS, change the access control lists to controller and controller in the end renew the flow tables of the network appliances so as to drop the packets. Thus there are numerous different implementations regarding the deploying topology of IDPS (figure 15).

To summarize, at least an IDS should be deployed in the controller. Any packet before entering the controller passes through the application firewall protecting the heart of SDN. Thus the controller's firewall will unencrypt the payload of each packet in order to apply through the IDS, granular or Deep Packet Inspection.

Now let's illustrate a SDN[29] architecture which supports a whole data center. The service provider (owner of the data center) will provide to tenants administrators the ability to configure a virtual firewall and a virtual IDS according to their needs. Both virtual IDPS and firewalls will be deployed, configured and maintained by the controller. Both preventive security systems have the purpose to protect the tenants VMs both from internet exposure (North-South Traffic) but also from the intranet flows (East-West Traffic) multi-tenant environment.

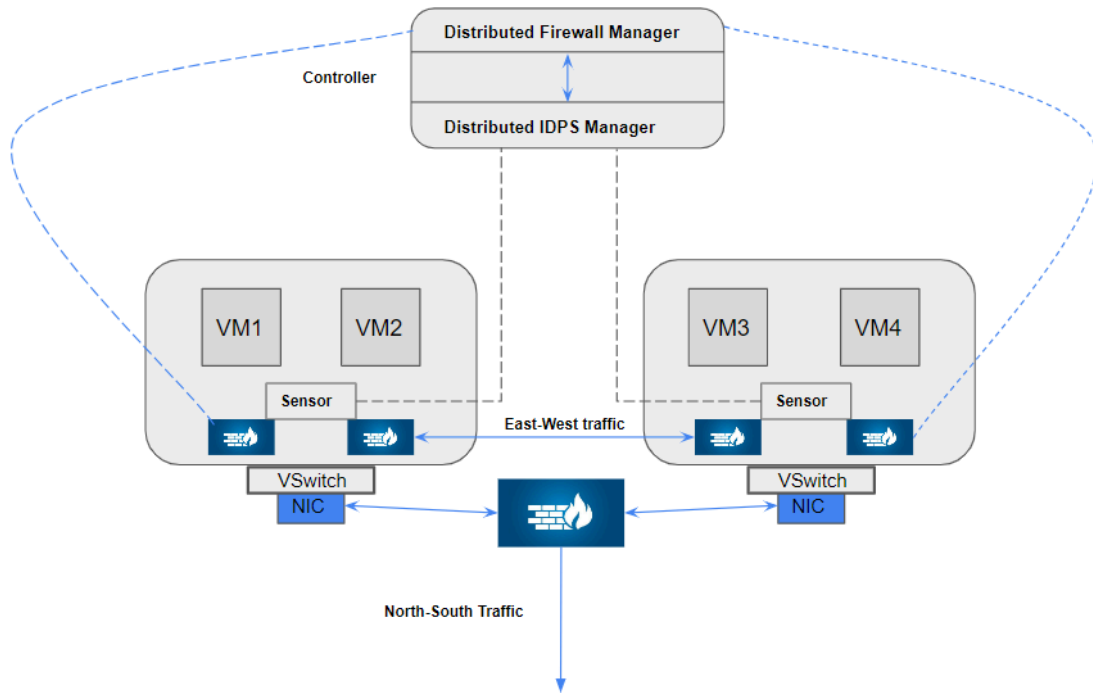


Figure 15, Distributed Firewall and IDPS manager within the controller

As it seems above, there is a distributed firewall [35] placed in the ingress and egress and in each VM. Behind each firewall there is an IDPS sensor/agent. The logs are sent immediately (live or mirroring) to the distributed IDPS manager (controller) to check their regularity. The IDPS manager interacts directly with the firewall manager in order to increase the resilience of the network. For example, If the IDS detects a ransomware in a payload will enable IPS to quarantine the packet and at the same time will send the IP source to the firewall so as to change the firewall rules and blacklist the malicious traffic.

Furthermore the contribution of PR-IDS (anomaly based) should be mentioned as an alternative, scalable and quite efficient detecting and prevention tool when it is deployed in SDN. Process-Related IDS is developed and works based on NSA (negative selection algorithm). It is a general assumption that every critical network prevention tool nowadays gets fed from a Cyber Threat Intelligence model so as to be aligned with the existing and advanced threats. PR-IDS is a tool that if it gets fed with the appropriate elements, given the threat vector and the environment where it is deployed can be very accurate.

At this point, It should be pointed out that in microsegmentation there is a barrier from theory to practice. Within a large virtual ecosystem with many VMs, as the number of VMs is increased so the complexity of implementing an effective IDPS is reducing. This happens because the mathematical calculations are increased exponentially and therefore there is a necessity for data reduction. Thus Artificial Intelligence and especially Machine Learning models are totally necessary for the functional implementation and not a luxury.

NSA is based on an artificial immune system and is inspired by a real (biological) immune system. The logic is as follows: We construct a set of data (S) which are considered intrinsic to the system and characterized as normal and based on this we construct a set of detectors (D) which recognize only the complement of S , for example non-intrinsic data , abnormal for the system. A detector if it detects even one element from the set S then it is automatically destroyed, otherwise it is written into the set D. This process is repeated until no detector recognizes any native element from the original set S. Then the detectors after being trained based on the initial data set are able to process new data and categorize it as intrinsic or non-intrinsic (abnormal).Therefore we notice that there are two phases, the training/creation of the detectors and in the second phase the detection/scanning of the new data. The data that detectors can be trained on can be numeric, alphanumeric, logic symbols as well as plain text. Rules must also be defined based on which detectors will be created and detected. Usually these rules are metric, i.e. mathematical models that measure distances, of course not in the usual sense of distance.

NSA version V-detector: The V-detector is an applied optimization of the NSA algorithm. V defines the word variable and defines the variability of the size of the set of detectors D. In the classical version of NSA the size of the set of detectors is fixed while in the V-detector an adjustment is made given the minimum distance from the data. Essentially we can imagine that by applying the V-detector the radius of the detector set is increased and the result is that it maximizes the coverage of the area between the samples of the original data and at the same time minimizes the detectors. Nevertheless, in order to apply the V-detector, it requires a certain processing of the data and of course the sample of the original data must be chosen very carefully.

Data: The data that originates or circulates within a network is numerous and comes from many different stages, such as external data (internet), databases, measurement results from sensors, etc. The pre-processing of the data consists of two stages: normalization and dimensional reduction. Data normalization basically adjusts the 5% and 95% percentages to -1 and 1 respectively, so that all the data is close to 0 which is necessary so that when dimensionality reduction is applied we do not reach the wrong conclusions. Reducing the dimension of the data is necessary to make the system much simpler and more efficient and of course faster. Data order reduction is performed through a statistical technique called PCA (principal component analysis). PCA is based on a k-dimensional orthogonal system and selects the dimensions based on the variances of the data in descending order.

3.4 Artificial Intelligence and its types

Nowadays, it's widely known the utilization of artificial intelligence and its subcategories (machine learning, deep learning, neural and convolutional networks) in almost all information technology fields. SDN is primarily based on software, thus by integrating intelligence in the software it is developed into clever and adjustable software which takes initiatives on its own. The advantages of Intelligent SD security systems are continuous end-to-end monitoring and awareness of the whole system, self healing properties and reconfiguration of the network architecture while an error or compromising event occurs[37]. Furthermore, intelligent software in conjunction with microsegmentation increases the resilience against APTs (advanced persistent threats) and its propagation. Last but not least, AI algorithms improve the efficiency, the speed and the accuracy of the classical defensive tools. For example neural and convolutional networks support the right development and implementation of Policies (analyzed above).

Additionally, CNN and NN integrate perfectly with IDPS by reducing exponentially the time of attack pattern recognition even in large datasets. As it mentioned in the microsegmentation section, the deployment of an IDPS is a NP problem. Thus the latter provides a defining solution to the problem of time management of applying an IDPS in microsegmentation[32]. Also CNN and NN can rapidly pinpoint a DDOS

attack and inform the monitor and remediation software to act as it is predefined by the administrators. Also NNs can live monitor the volume and the rate of traffic in a SDN, predict the congestion in some nodes and promptly find alternative paths to send/divide the traffic. Thus NNs can act as load balancers and prevent single point of failure disruption on controllers and in the infrastructure layer. Generally speaking the goal of unsupervised learning is focusing on data exploration and pattern discovery. Thus, unsupervised learning in the cybersecurity field is necessary and brings back great profit and effects, especially in IDPS and threat analysis. A very known and efficient machine learning algorithm applied on unsupervised data is the K-Means.

The K-means algorithm is described by 5 different steps. The logic is quite simple and based on statistical models and repetition, however there are many computations thus it's used a variant approach(e.g Lloyd's method). The K-means algorithm significantly improves the efficiency and the precision of the IDPS systems If we succeed to pass all the data from a data reduction technique before[26]. The classification of data can be utilized by the policy enforcer, network flows and can lead to data reduction. The machine learning models are necessary in order to reduce the data before passing through the IDPS, otherwise the delay will be much bigger.

AI and ML models analyze in each system the behavior of the machine in conjunction with its users, create some boundaries given the "local" data and after establishing the red zones/ new boundaries. These red zones define the acceptable behaviors in the system. This is the most common way IDS exploit AI in order to increase their effectiveness based on anomaly based detection. Through the latter procedure AI also reduces the wrong alerts(False positive, False negative) in the IDS.

It is worth mentioning that SDN is employed by huge organizations which have placed SCADA(Supervisory Control and Data Acquisition) systems to monitor and control processes and operations for their whole ecosystem. K-Means is the most appropriate algorithm for classifying large amounts of unsupervised data.

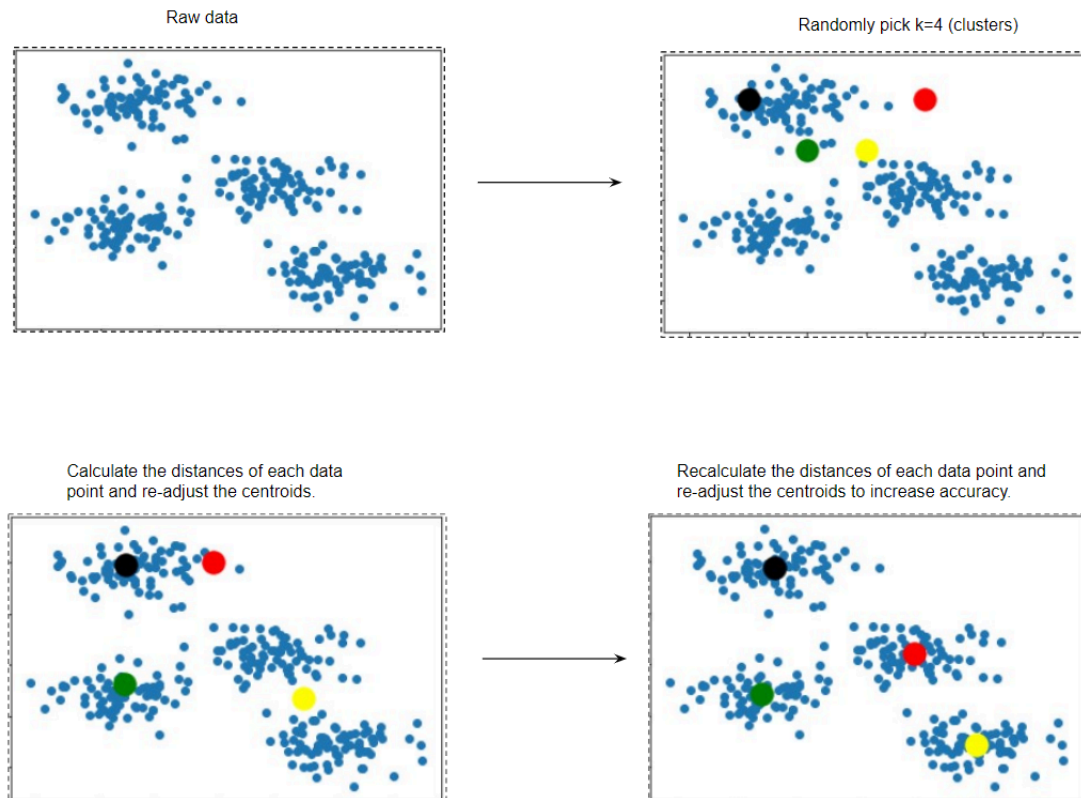


Figure 16, Representation of Implementation of K-means

Within SCADA systems are used the autoencoders. Autoencoders are very useful in SCADA systems because their data usually have specific forms. Autoencoders belong to the broader category of artificial neural networks and have many applications on unsupervised data, one of the main ones being their use in IDS systems. There are 3 stages of an autoencoder: The encoder, the code and the decoder. The encoder takes as input data and essentially compresses the data and encodes it. The decoder processes the code, which is the processed information that the encoder has compressed, and learns not to recreate the original data using the code. The result is that the input data is very similar to what is being interpreted. The encoder discards the useless information/noise and builds spatial representations (based on dominant features) of the original data.

How can a Model-Based IDS take advantage of autoencoders?

We train autoencoders based on normal/daily data processed by an IDS. This means that all the normal data is present in the processed code and can be reproduced by the

decoder at any time. So when an abnormal data is produced by the decoder there will be a large error between input and output and thus the algorithm understands that the data is malicious.

3.5 Moving Target Defense Mechanisms

The nature of cloud and virtual machines are described by static elements. This stability serves the engineers and the administrators and makes their daily work simpler. However, this situation lurks many dangers because all the phases of the lifecycle of an attack can be deployed easier. Moving Target Defense(MTD) actions target to deceive the opponent in many different fields(strategy games, real war, etc). Thus, practically speaking MTD [36] is a proactive cybersecurity strategy that aims to increase the complexity and uncertainty for attackers by constantly changing the attack surface (open ports, Operating systems, software versions and other static elements which can be exploited) and system configuration.

MTD defending strategies are based on two basic categories. These techniques are deployed based on clever security models or these which just change the IP packet information(from Layer 2 to Layer 7) .

Most known obfuscating techniques [1]:

Shuffling technique: As we can understand from the title, shuffling is a modification of the order of some entities in order to stop and also lure the attacker. The implementation of this technique varies and depends on the administrator's desires. For example when a server detects malicious alerts from one or more users, automatically internally redirects them to a honeypot server (figure 17) with similar attack surface as the previous. This move will give more time to the blue/red team to act and collect further information for the attackers and the purpose of the attack. The MTD mechanism usually is deployed in the controller and has at least 3 basic components, the MTD network model, a computation engine and a collector of packets.The network model analyzes packets from collector and when it finds a

suspicious reconnaissance packet, it modifies it in order to confuse the attacker and increase his workload.

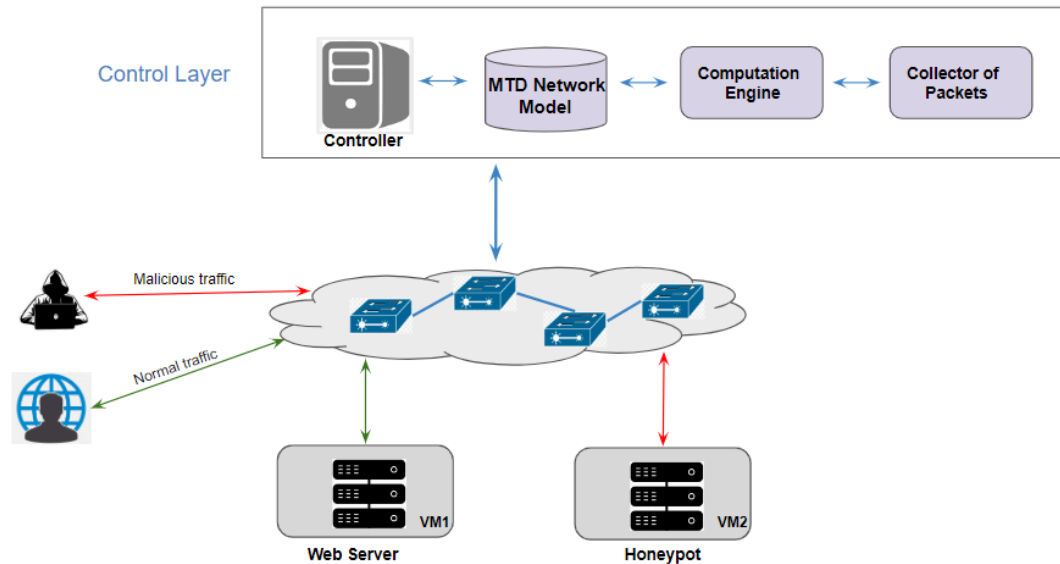


Figure 17, MTD model redirect malicious user to a honeypot server

Diversification technique: The attackers by using many state of the art side channel attacks or code reuse attacks succeed in analyzing the binaries and find similarities between them which reveal [30] the version of software. After the attackers create a payload to exploit the vulnerability of this specific version.

The developers in order to face these kinds of attacks create a binary generator which diversifies the basic binary for each different client request. As a consequence attackers can not reach safe conclusions by analyzing the binaries.

Increasing Redundancy: Increasing redundancy of the basic resources of important resources and network nodes are totally essential. Cybersecurity engineers tend to add extra network appliances in all the perimeter of the network in order to better balance the volume of traffic and to increase the resilience against DDOS. Another common best practice and strategy is to place many different proxies inside and outside of the network so as to conceal from the users and possible attackers the real IPs and the attack surface of the vital components. Also when the IDS detects a malicious activity, after blocking the threat there is a shuffle mechanism between these proxies

in order to prevent any future potential similar attack. Last but not least we could mention/add the migration technique here. When the controller is notified by the defensive tools for an active threat within a VM can rapidly decide to migrate the applications from the contaminated VM to another new VM. This migration is implemented based on the capacity of the VM, the network bandwidth and the security tag of the VM. Sometimes many VMs hosting important services or data are automatically migrated proactively to reduce the risk of compromise.

IP constant mutation: SDN recognizes all the valuable hosts within the network and constantly changes their IPs. [30] Each host has a real IP which is unique and does not change and a virtual IP which changes regularly. The host can see only the real IP but anybody who tries from the internet to reach the host sees the virtual IP. The swap between the virtual and the real IP takes place at the gateway. Of course the whole procedure is orchestrated from the controller and the significant is that the performance of the network is characterized by stability. This is a known and very efficient technique in SDN and it's employed to defend against any kind of information gathering attacks. Its efficiency surpasses 98.8%.

Computations limit and mathematics behind MTD: By reading about the elaborate MTD techniques the average reader should pose many questions. How does the background of MTD work? Which are the principles and the foundations of these mechanisms? Is MTD used also in other areas than the information field? The foundation of MTD mechanisms is game theory, discrete mathematics, theory of graphs, linear and non-linear algebra and Stochastic Modelling. Even the most simple MTD mechanism will combine the definition of equilibrium, many rearrangements of graphs and the Markov chain. As it is perceived the blend of the above fields brings exponential computations even for the simplest implementation.

Conclusion: MTD mechanisms are considered state of the art and unique solutions to face the stability of cloud infrastructure. Synchronous MTD mechanisms take advantage of all the above techniques and create a combination of obfuscated techniques to muddy the waters for the attackers. However, the implementation of all these techniques has a ceiling because of the complex mathematical computations. At the moment, a well established controller in conjunction with attack graphs and well

defined MTD mechanisms can not support more than one thousand nodes. Thus, when MTD solutions are applied in a large network, there should exist many controllers and a parallel multi- MTD mechanism integrated into the control layer.

It's important to note that while moving target defense can provide additional layers of defense and deterrence, it is not a silver bullet and should be complemented with other cybersecurity practices, including secure coding, vulnerability management, and incident response planning. The implementation of this technique varies and depends on the administrator's desires.

Chapter 4: Security challenges arise from the deployment of SDN in IOT

4.1 General description

As it has been already referred above the SDN architecture can integrate with IOT and scale up the efficiency of each ecosystem. However, both IoT and SDN have their individual security challenges[32], the combination of the two can introduce additional security risks. Here are some reasons why IoT may be less secure in an SDN environment:

Scale and Complexity: IoT networks can consist of thousands or even millions of devices, each with its own unique set of security requirements and vulnerabilities. When integrated with SDN, managing and securing such a large-scale and complex ecosystem becomes more challenging, especially as the number of IoT devices continues to grow rapidly [26].

Heterogeneity: IoT devices come from various manufacturers and often use different communication protocols and standards. Integrating them into an SDN environment requires interoperability and adaptation, which can introduce vulnerabilities if not properly implemented. Compatibility issues and security gaps may arise due to the diverse nature of IoT devices.

Limited Computing and Storage Resources: Many IoT devices have limited computing power and storage capabilities, which can hinder the implementation of robust security measures. Encryption, authentication, and other security mechanisms may be resource-intensive, making it difficult to enforce strong security controls on IoT devices within an SDN architecture.

Lack of Standardization: The IoT industry lacks comprehensive security standards and protocols. Different manufacturers may implement varying security measures or overlook security considerations altogether. When these devices are integrated into an SDN, the lack of standardized security practices can create inconsistencies and vulnerabilities.

Communication Channels: IoT devices often rely on wireless communication technologies such as Wi-Fi, Bluetooth, or cellular networks. These communication channels are susceptible to eavesdropping, spoofing, and other attacks. Integrating IoT devices with SDN requires secure communication channels to protect the data transmitted between devices and the central SDN controller.

Firmware and Software Vulnerabilities: IoT devices often have firmware and software that may contain security vulnerabilities. In many cases, these vulnerabilities go unnoticed or unpatched for extended periods. Integrating such devices into an SDN environment increases the attack surface and potential impact of these vulnerabilities.

Lack of Security Updates and Patches: IoT devices may not receive regular security updates or patches from manufacturers. Without timely updates, vulnerabilities in IoT devices can remain unaddressed, making them more susceptible to exploitation within an SDN architecture.

To enhance IoT security in an SDN environment, it is crucial to implement several measures, such as including strong device authentication, encryption, intrusion detection systems, regular security audits, and robust access controls. Additionally, collaboration among IoT device manufacturers, SDN vendors, and security experts is essential to establish industry-wide security standards and best practices.

4.2 Manufacturer Usage Description (MUD) solution

Specifically, nowadays a totally new model/framework is developed in order to increase the resilience of IOT as an entity and in conjunction with SDN technology. The basic defensive strategy is based on the integration of machine learning models and Manufacturer Usage Description (MUD) in IDPS.

Manufacturer Usage Description is a totally new concept and its purpose is to increase the degraded security of IOT. MUD is an embedded software on new IOT devices and supports advertising the operational specifications and the intended usage such as

dedicated ports, communication protocols, policies and communication patterns that are necessary for the utilization of each device.

Each time an IOT device connects to a new network sends a predetermined URL to the MUD manager. The MUD manager communicates with the MUD server who sends the instructions file for the device. After, the MUD manager receives the file and within the instruction file there is the access control list for the current device. The MUD manager authenticates devices identity and then enforces the access control list on the local network/switch(L2/L3). With this action, the MUD concept diminishes the big landscape of threats and vulnerabilities for IOT and increases the security.

How MUD is adjusted to SDN architecture

MUD takes place in the SDN architecture through 2 basic components. The first is, the MUD policy engine interacts with the application plane and the controller at the same time (does similar work with the MUD manager)[38]. The MUD policy engine communicates with the MUD file server and receives (figure 18) the description of each IOT device and also stores all the discovered IOT devices. The MUD policy engine analyzes the profiles and dynamically modifies the network flows. However even in this case there are spoofing attacks which have to be countered. Thus, the second component MUD collector is placed between the controller and the northbound interface and in conjunction with an anomaly detector gathers information from the switches(L2/L3) and does granular analysis to recognize If the existing attributes of each IOT device are aligned and authenticated by the policy engine.

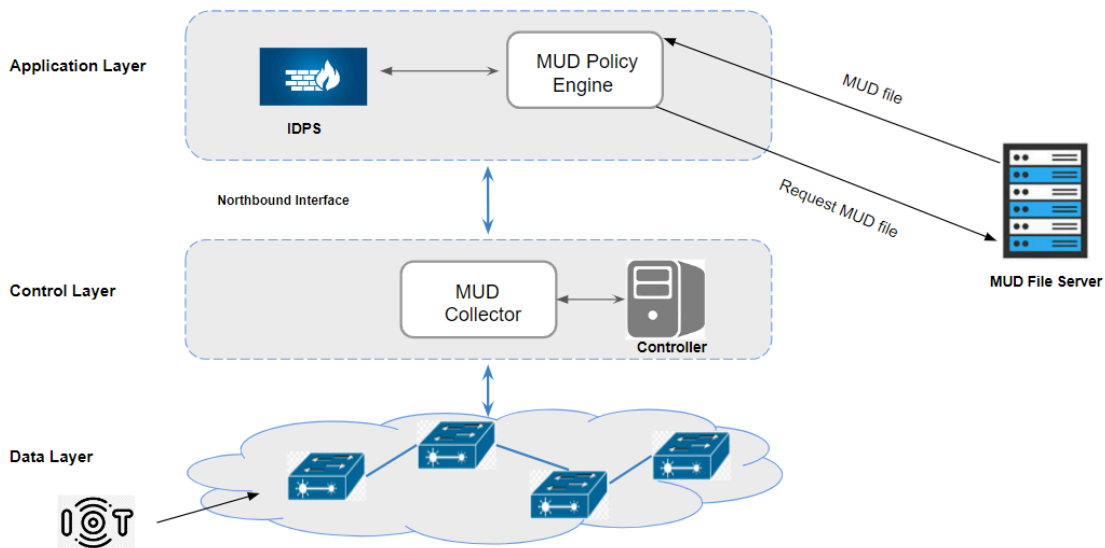


Figure 18, Deploying MUD technology into the control layer.

4.3 Secure IOT through the combination of MUD with ML

ML models takes advantage of MUD information

The last but awesome usage of the MUD concept is its interaction with machine learning models. As we already know the Machine and Deep learning are inevitable If we want to create an efficient IDPS [26]. The subfield of ML of all supervised or semi-supervised models gains a huge advantage from the MUD instruction file. The instruction file of each device feeds the above-mentioned ML models with specific information and as a consequence it does increase the efficiency and the accuracy of the model.

The contribution of MUD in IOT security landscape

By leveraging MUD within an SDN environment, organizations can enhance the security of their IoT deployments by implementing granular access controls, enforcing security policies, detecting anomalies, and improving network visibility. MUD provides a standardized approach to securely integrate and manage IoT devices within an SDN architecture, strengthening the overall security of the IoT security landscape.

Research showed that by feeding the IDPS with MUD information can increase the detection rate TCP attacks by at least 15%(from 80% to 95%) and the UDP attacks by 12%(from 78% to 90%). As it seems the MUD concept brings great results and value to the degraded IOT security landscape.

By combining MUD, ML, DL, and IDPS, organizations can enhance the security of their IoT deployments in SDN environments. These technologies collectively contribute to access control, policy enforcement, threat detection, anomaly detection, network segmentation, and automated response mechanisms. Together, they provide a comprehensive and proactive approach to securing IoT devices, mitigating risks, and protecting critical infrastructure in the rapidly expanding IoT landscape within SDN context.

Chapter 5: Conclusions and the future of the SDN security

Generally speaking, all the existing security challenges of the conventional networks constitute a threat also for software defined networks. Furthermore, the new supporting features of SDN, the interaction between controllers and L2/L3 switches from different providers and SDN technology used for SD-Lan and SD-Wan solutions, pave the way for new unknown security challenges. Protecting the CIA of controllers and the trust relationship between third party applications and controllers pose another vital challenge. Another existing weakness, lies in large SDN infrastructure and specifically in the communication between simple controllers and higher level controllers or application controllers. Last but not least, assuring the interoperability and the security of the existing legacy protocols before integrating them within SDN technology is not an easy task.

As it was stated above, all the common attacks in conventional networks are applied also in all layers of software defined networks. Starting from the data layer, the most common attacks are the man-in-the middle attack (MitM), the border gateway protocol (BGP) route hijacking attacks and the eavesdropping/sniffing attacks. Continuing to the control layer, it was mentioned the link fabrication attack, the flow table overloading and numerous internal and external DDOS attacks launching in different ways. The application plane is threatened by the reflected, stored or blind Cross-Site Scripting attack (XSS), SQL injection attack and the CSRF attacks. The most harmful impacts are, non-availability of switches and controllers, exfiltration of sensitive data and cyber espionage which can result even in global internet instability.

With the purpose of applying a holistic strategy against cyber attacks, it is essential to develop a multidimensional defensive strategy based on cutting edge technologies. The first defensive dimension are both high and low level policies. By analyzing the traffic metadata are often defined new policies which drastically reduce the complexity of network flows and increase generally the efficiency and the security of the flows and the consistency of the whole SDN architecture. The second defensive countermeasure is the power of distributed firewalls (DF) and microsegmentation.

The DF developed beyond the edges of the perimeter, within the SDN virtualized environments and totally increased the internal protection, the defense against APT and the most amazing feature is the topology independence and its advantages. Moreover DF feeded by high level policies can increase the performance and security of the whole environment. The third countermeasure is a multi layer IPS, deployed in many different locations, from basic focal points of perimeter ingress and egress(network-based) to endpoint machines(host-based).

Policies, distributed firewall and IPS have to manage a massive volume of traffic and logs and it's inevitable to work without implementing machine learning algorithms so as to supervise data and more precisely to apply a data reduction or data classification. Thus, the fourth defensive countermeasure are artificial intelligence and machine learning models, such as k-means which can by far increase the performance and the consistency of the other cutting edge defensive solutions. The fifth and last countermeasure is leveraging the power of moving target defense mechanisms (MTD). MTD acts proactively within the environment and through shuffling many static elements as IPs, increases the uncertainty and complexity for potential attackers. MTD constitutes one of the most cutting edge technologies defensive solutions in cybersecurity.

Worth mentioning is that many challenges have occurred from the deployment of SDN architecture within the IOT community. These challenges derive from the heterogeneity, the limited computing and storage resources and the lack of standardization of IOT devices. One solution to reduce the vulnerabilities landscape of IOT is to take advantage of manufacturer usage description (MUD). After deploying the MUD solution within a SDN architecture, it is defined according to each IOT device which behavior is normal and increases the visibility and the resilience of the SDN controller.

As we traverse the landscape of Software-Defined Networking (SDN), this thesis has meticulously explored the current state of SDN security and a holistic defense strategy of SDN deployed in 5 basic axes . Our journey through the realms of SDN security has described the basic architecture, uncovered challenges, attacks on all layers (data, control, application) and promising solutions, providing a nuanced understanding of

the evolving paradigm. As we conclude this exploration, it is evident that the future of security in SDN demands a proactive and adaptive approach, one that is not only cognizant of current threats but anticipates the dynamic landscape that lies ahead. Thus we should take into account two different facts so as to describe the future of SDN security and its evolution

The first fact is that there is a huge growing market which is day by day moving forward to SDN based network architectures; and around 2032 the overwhelming majority of tech giants and ISPs will use exclusively SDN technology. Thus the rapid evolution and development of Software Defined Networks is a one way street and as it is widely known while software is developed under the pressure of time more vulnerabilities tend to occur.

The second fact is that cybercriminals and hacktivists groups know about the rapid development of SDN and in parallel they develop new techniques and optimize the existing so as to take advantage of any misconfiguration which will occur as a consequence of the rapid development of SDN.

As it is understandable the security of SDN is not a static endeavor but a dynamic process that must evolve in tandem with technological advancements and emerging threats. A holistic approach that encompasses standardized security practices, adaptive defense based on multi-dimensional mechanisms, and ongoing collaboration between the providers of SDN solutions is essential to secure the future of SDN.

Bibliography

[1] GeeksforGeeks, <https://www.geeksforgeeks.org/software-defined-networking>, accessed May 2023,

[2] Digi, <https://www.digi.com/blog/post/software-defined-networking>, accessed May 2023

[3] Nutanix, <https://www.nutanix.com/info/software-defined-networking>, accessed June 2023

[4] Techtarget, <https://www.techtarget.com/searchnetworking/definition/SDN-controller-software-defined-networking-controller>, accessed June 2023

[5] GeeksforGeeks, <https://www.geeksforgeeks.org/difference-between-control-plane-and-data-plane/>, accessed June 2023

[6] Open Networking Foundation, OpenFlow Switch Specification, 2012, <https://www.geeksforgeeks.org/difference-between-control-plane-and-data-plane>, accessed March 28, 2023

[7] Z. A. Bhuiyan, S. Islam, Md. M. Islam, A. Ullah, F. Naz, and M. S. Rahman, On the (in)Security of the Control Plane of SDN Architecture: A Survey, <https://doi.org/10.1109/access.2023.3307467>, accessed June 2023

[8] A. L. Aliyu, A. Aneiba, M. Patwary, and P. Bull, "A trust management framework for Software Defined Network (SDN) controller and network applications, <https://www.sciencedirect.com/science/article/abs/pii/S1389128620311105?via%3Dihub>, accessed July 2023

- [9] Techtarget,
<https://www.techtarget.com/searchnetworking/definition/software-defined-networking-SDN>, accessed July 2023
- [10] E. Sakic, F. Sardis, J. W. Guck, and W. Kellerer, Towards adaptive state consistency in distributed SDN control plane,
<https://ieeexplore.ieee.org/document/7997164>, accessed July 2023
- [11] SDxCentralStudios,
<https://www.sdxcentral.com/networking/sdn/definitions/what-the-definition-of-software-re-defined-networking-sdn/>, accessed July 25, 2023
- [12] BMC , <https://www.bmc.com/blogs/software-defined-networking>, accessed July 2023
- [13] K. Finley,
<https://www.wired.com/2015/11/trade-pact-could-bar-governments-from-auditing-source-code/> accessed Sep. 25, 2023
- [14] Open Networking Foundation, “Principles and Practices for Securing Software-Defined Networks,”
https://opennetworking.org/wp-content/uploads/2014/10/Principles_and_Practices_for_Securing_Software-Defined_Networks_applied_to_OFv1.3.4_V1.0.pdf, accessed June 23, 2023
- [15] R. R. Stewart, M. Dalal, and A. Ramaiah, “Improving TCP’s Robustness to Blind In-Window Attacks,” <https://datatracker.ietf.org/doc/rfc5961/>, accessed Sep. 25, 2023
- [16] Techtarget,
<https://www.techtarget.com/iotagenda/definition/man-in-the-middle-attack-MitM>, accessed September. 28, 2023
- [17] Kadiska,
<https://kadiska.com/what-is-bgp-hijacking-how-to-detect-and-prevent-it/>, accessed Sep. 31, 2023

- [18] TechTarget, <https://www.techtarget.com/searchsecurity/tip/How-does-BGP-hijacking-work-and-what-are-the-risks>, accessed September 13, 2023
- [19] Investopedia, <https://www.investopedia.com/terms/e/eavesdropping-attack.asp>, accessed September 23, 2023
- [20] OWASP, <https://owasp.org/www-community/attacks/xss/>, accessed
- [21] OWASP, https://owasp.org/www-community/attacks/SQL_Injection, accessed March 28, 2023
- [22] OWASP, <https://owasp.org/www-community/attacks/csrf>, accessed March 28, 2023
- [23] Abdunasser Alowa, Thomas G Fevens, Yaser Khamayseh, Survival backup strategy for controller placement problem in Software Defined Networking, https://www.researchgate.net/publication/357446898_Survival_backup_strategy_for_controller_placement_problem_in_Software_Defined_Networking, accessed April 22, 2023
- [24] S. Shin and G. Gu, “Attacking software-defined networks: A first feasibility study,” https://www.researchgate.net/publication/262154123_Attacking_software-defined_networks_A_first_feasibility_study, accessed May 12, 2023
- [25] S. Khan, M. A. Bagiwa, A. W. A. Wahab, A. Gani, and A. Abdelaziz, “Understanding link fabrication attack in software defined networks using formal methods,” https://www.researchgate.net/publication/341317420_Understanding_Link_Fabrication_Attack_in_Software_Defined_Network_using_Formal_Methods, accessed May 29, 2023
- [26] Dijiang Huang, Ankur Chowdhary, Sandeep Pisharody, Software-Defined Networking and Security, CRC press, 2018

- [27] Leo Caldarola, Amine Choukir, High Level Policies in SDN, https://www.researchgate.net/publication/315544868_High_Level_Policies_in_SDN, accessed May 02, 2023
- [28] Mohamed Rahouti, Kaiqi Xiong, Moussa Ayyash, SDN Security Review: Threat Taxonomy, Implications and Open Challenges, <https://ieeexplore.ieee.org/document/9760465>, accessed May 11, 2023
- [29] Anass Sebbar, Karim ZKIK, Youssef Baddi, Using advanced detection and prevention technique to mitigate threats in SDN architecture, <https://ieeexplore.ieee.org/document/8766552>, accessed Jul 29, 2024
- [30] Jafar Haadi, Ehab Al-Shaer, OpenFlow Random Host Mutation: Transparent Moving Target Defense using Software Defined Networking, <https://dl.acm.org/doi/10.1145/2342441.2342467>, accessed June 12, 2023
- [31] Nam-Thang Hoang, Hai-Nam Nguyen, Hai-Anh Tran, A Novel Adaptive East–West Interface for a Heterogeneous and Distributed SDN Network, https://www.researchgate.net/publication/359406099_A_Novel_Adaptive_East-West_Interface_for_a_Heterogeneous_and_Distributed_SDN_Network, accessed August 12, 2023
- [32] Setiadi Yazid, Mochamad Teguh Kurniawan, A Systematic Literature Review of Security Software Defined Network: Research Trends, Threat, Attack, Detect, Mitigate, and Countermeasure, https://www.researchgate.net/publication/338520125_A_systematic_literature_review_of_security_software_defined_network_research_trends_threat_attack_detect_mitigate_and_countermeasure, accessed august 02, 2023
- [33] Vaibhav Hemant Dixit, Sukwha Kyung, Ziming Zhao, Challenges and Preparedness of SDN-based Firewalls, https://www.researchgate.net/publication/323790568_Challenges_and_Preparedness_of_SDN-based_Firewalls, accessed July 25, 2023

[34] Tao Han, Syed Rooh Ullah Jan, Zhiyuan Tan, A comprehensive survey of security threats and their mitigation techniques for next-generation SDN controllers,https://www.researchgate.net/publication/331968112_A_Comprehensive_Survey_of_Security_Threats_and_their_Mitigation_Techniques_for_next-generation_SDN_Controllers, accessed June 29, 2023

[35] Ozgur Yurekten, Mehmet Demirci, Cyber threat intelligence assisted defense system for software-defined networks,<https://www.sciencedirect.com/science/article/abs/pii/S1389128621001262>, accessed June 05, 2023

[36] Saamil Macwan, Chung-Horng Lung, Investigation of Moving Target Defense Technique to Prevent Poisoning Attacks in SDN, <https://ieeexplore.ieee.org/document/8817213>, accessed June 25, 2023

[37] Majd Latah, Levent Toker, Artificial intelligence enabled software-defined networking: a comprehensive overview,<https://ietresearch.onlinelibrary.wiley.com/doi/full/10.1049/iet-net.2018.5082>, accessed July 22, 2023

[38] Rania A. Elsayed, Reem A. Hamada, Shaimaa Ahmed Elsaid, Securing IoT and SDN systems using deep-learning based automatic intrusion detection,<https://www.sciencedirect.com/science/article/pii/S2090447923001004>, accessed April 25, 2023

[39] Precedence Search,, accessed September 23, 2023 <https://www.precedenceresearch.com/software-defined-data-center-market>,

[40] Geeksforgeeks, <https://www.geeksforgeeks.org/software-defined-networking>, accessed September 17, 2023

[41] Analytics Vidhya, accessed September 23 2023, <https://www.analyticsvidhya.com/blog/2019/08/comprehensive-guide-k-means-clustering/>