Prevention of Eavesdropping in SDN using IP Encryption

Abstract:

In the realm of Software-Defined Networking (SDN), the need for robust security measures to protect data and communications has become paramount. This report details a project that addresses the challenge of safeguarding IP addresses within an SDN infrastructure to prevent passive attacks. We propose a novel approach that employs encryption algorithms at the first switch in the network, followed by decryption algorithms at the last switch before data exits the network.

The project centers around a core SDN architecture consisting of a central controller and multiple switches. Our objective is to ensure end-to-end security by encrypting IP addresses at the network ingress and decrypting them at the egress, thus safeguarding critical information from passive attacks.

The report discusses the selection and implementation of encryption and decryption algorithms, their integration into the SDN architecture, and the impact on network performance. Additionally, we evaluate the effectiveness of this solution in preventing passive attacks and maintaining the confidentiality of IP addresses.

By adopting this approach, we aim to enhance the security posture of SDN networks, mitigate potential vulnerabilities, and protect sensitive IP address information throughout its traversal within the network. This report provides insights into the practical implementation of this security mechanism and its potential implications for future SDN deployments.

Introduction:

In the ever-evolving landscape of modern networking, Software-Defined Networking (SDN) has emerged as a transformative paradigm, offering dynamic control, flexibility, and scalability in managing network resources. As organizations increasingly rely on SDN to optimize network performance, they also face the imperative challenge of safeguarding sensitive data and communications from potential threats. Passive attacks, such as eavesdropping and data interception, pose a substantial risk to the confidentiality of critical information traversing SDN networks, especially in the context of IP address exposure.

This project endeavors to address this critical security concern by proposing a novel approach: the implementation of encryption algorithms at the first switch within the SDN network infrastructure, and corresponding decryption algorithms at the final switch before data exits the network. By doing so, we aim to establish end-to-end IP address encryption to thwart passive attacks, ensuring the confidentiality and integrity of IP address information.

In the following sections, this report will delve into the intricate details of our project, which spans the selection and integration of encryption and decryption algorithms, the incorporation of this security mechanism within the SDN architecture, and a comprehensive evaluation of its impact on network performance. The underlying objective is to enhance the security posture of SDN networks, mitigating potential vulnerabilities and ensuring the protection of sensitive IP address data throughout its traversal within the network.

Background:

1) Software Defined Networking (SDN):

SDN stands for Software Defined Network which is a networking architecture approach. It enables the control and management of the network using software applications. Through Software Defined Network (SDN) networking behavior of the entire network and its devices are programmed in a centrally controlled manner through software applications using open APIs.

To understand software-defined networks, we need to understand the various planes involved in networking.

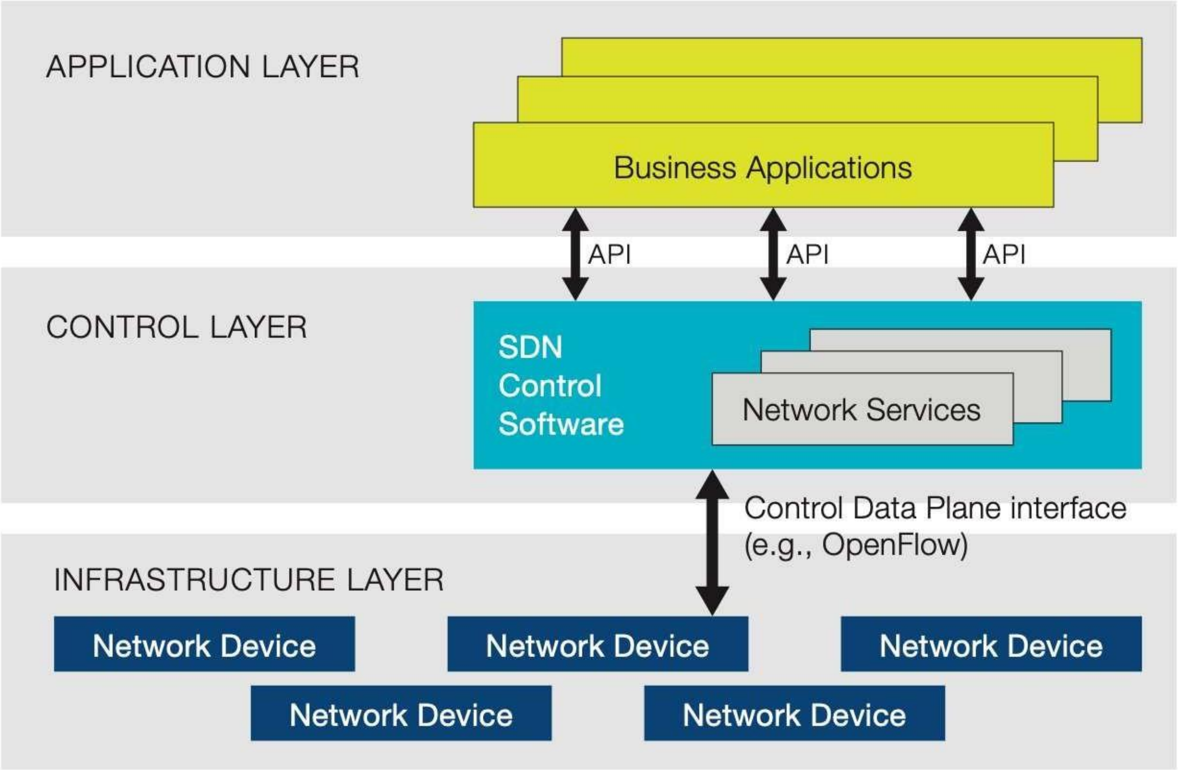
Data plane: All the activities involving as well as resulting from data packets sent by the end-user belong to this plane. This includes:

1. Forwarding of packets.
2. Segmentation and reassembly of data.
3. Replication of packets for multicasting.

Control plane: All activities necessary to perform data plane activities but do not involve end-user data packets belong to this plane. In other words, this is the brain of the network. The activities of the control plane include:

1. Making routing tables.
2. Setting packet handling policies.

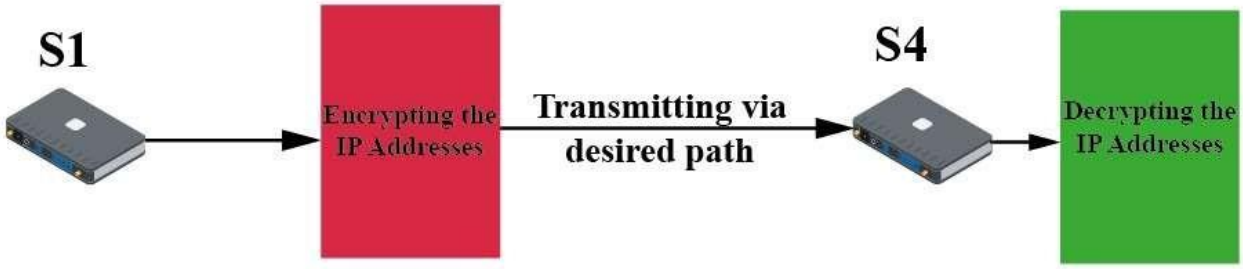
In short, it can be said that- SDN acts as a “Bigger Umbrella or a HUB” where the rest of other networking technologies come and sit under that umbrella and get merged with another platform to bring out the best of the best outcome by decreasing the traffic rate and by increasing the efficiency of data flow. The architecture of SDN is highlighted below:



2) <u>Mininet:</u>

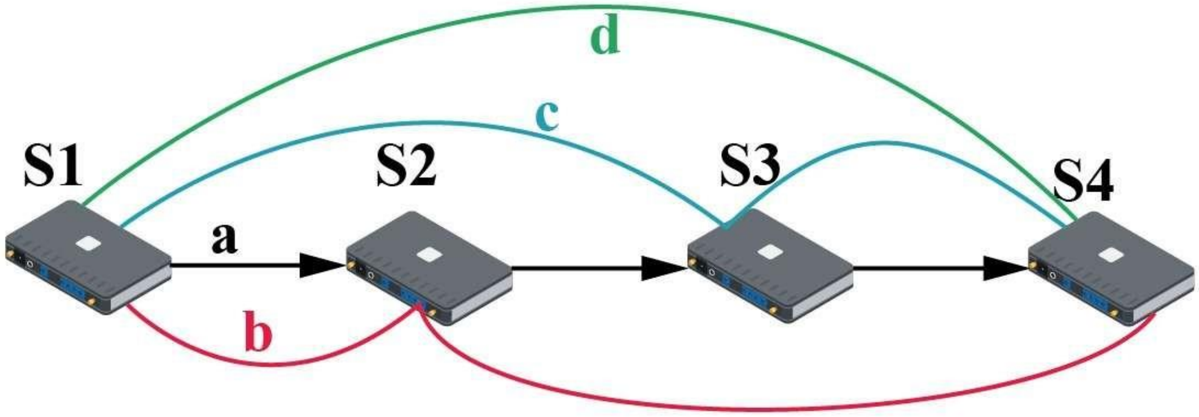
Mininet is a powerful network emulator that allows users to create virtual networks on a single machine, enabling the development, testing, and simulation of Software-Defined Networking (SDN) environments. Built on lightweight Linux containers, Mininet can simulate a complete network of hosts, switches, controllers, and links, all running real-world kernel code. This makes it ideal for SDN research and development, as it supports OpenFlow protocols and integrates easily with SDN controllers like OpenDaylight, Floodlight, and Ryu. Mininet is highly extensible, providing a command-line interface and Python API for custom network topologies and testing scenarios, making it widely used in both academic research and industry to prototype and validate network configurations and applications before deployment.

Detailed Project Outline:


S1
Encrypting the IP Addresses
Transmitting via desired path
S4
Decrypting the IP Addresses

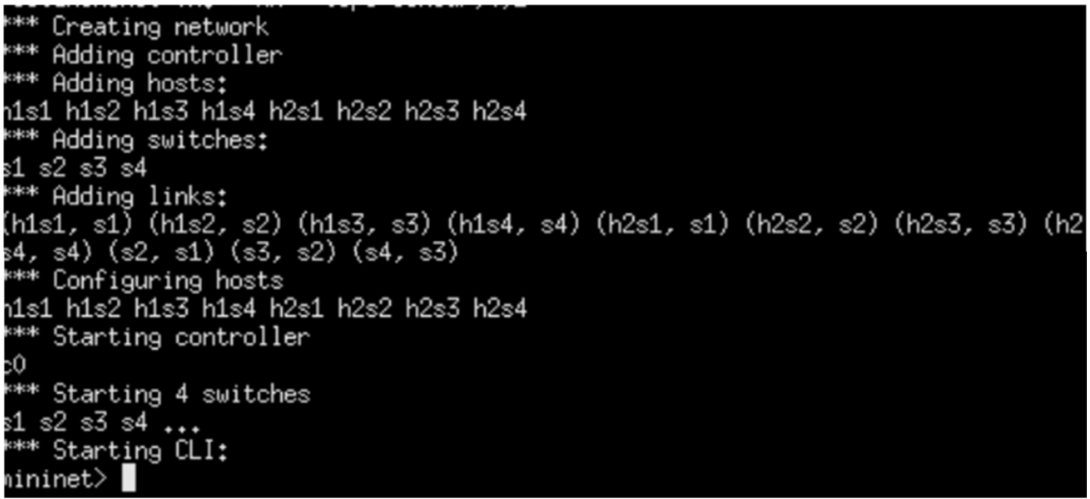
The project aims at developing a more secured and reliable mode of transmission of data flow from source ip to destination ip, without having to fear about passive attacks such as eavesdropping. The system specifications could be visualized as in the figure above.

The algorithm encrypts the src and dest ip_addresses, and randomly allocates a path from (a-d) to the packet for transmission throughout the network. The paths set are depicted in the figure below:

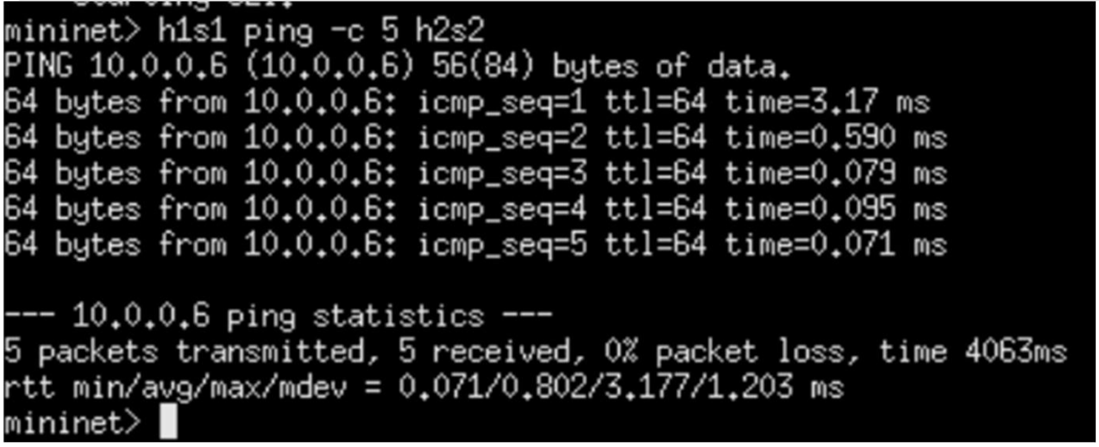


The topology comprises of four openflow switches, pox controller and eight hosts. The working of the code and the creation of topologies has been depicted in the coming section.

Creation of SDN topology using Mininet:

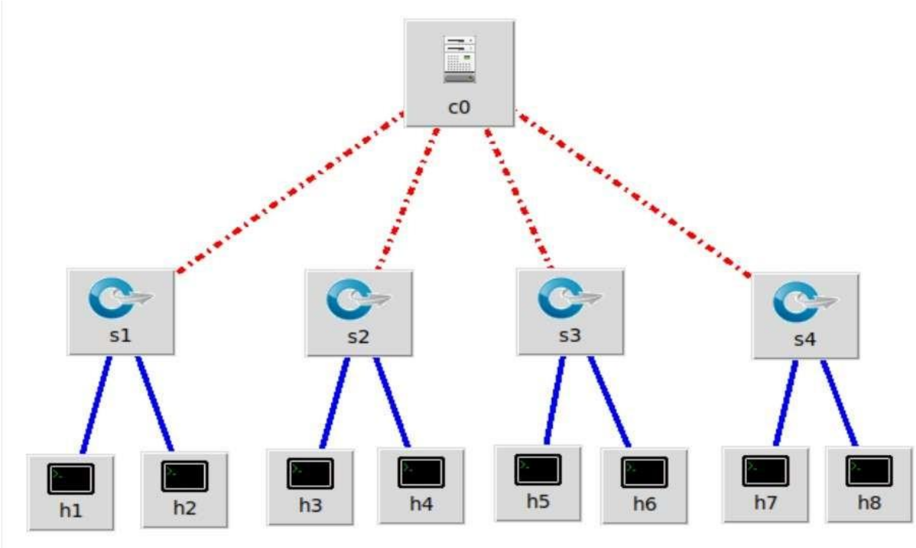


```
*** Creating network
*** Adding controller
*** Adding hosts:
h1s1 h1s2 h1s3 h1s4 h2s1 h2s2 h2s3 h2s4
*** Adding switches:
s1 s2 s3 s4
*** Adding links:
(h1s1, s1) (h1s2, s2) (h1s3, s3) (h1s4, s4) (h2s1, s1) (h2s2, s2) (h2s3, s3) (h2
s4, s4) (s2, s1) (s3, s2) (s4, s3)
*** Configuring hosts
h1s1 h1s2 h1s3 h1s4 h2s1 h2s2 h2s3 h2s4
*** Starting controller
c0
*** Starting 4 switches
s1 s2 s3 s4 ...
*** Starting CLI:
mininet> █
```



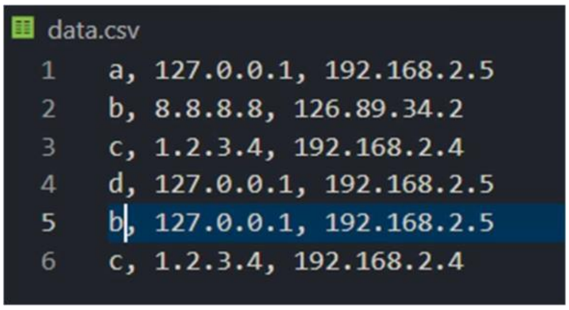
```
mininet> h1s1 ping -c 5 h2s2
PING 10.0.0.6 (10.0.0.6) 56(84) bytes of data.
64 bytes from 10.0.0.6: icmp_seq=1 ttl=64 time=3.17 ms
64 bytes from 10.0.0.6: icmp_seq=2 ttl=64 time=0.590 ms
64 bytes from 10.0.0.6: icmp_seq=3 ttl=64 time=0.079 ms
64 bytes from 10.0.0.6: icmp_seq=4 ttl=64 time=0.095 ms
64 bytes from 10.0.0.6: icmp_seq=5 ttl=64 time=0.071 ms

--- 10.0.0.6 ping statistics ---
5 packets transmitted, 5 received, 0% packet loss, time 4063ms
rtt min/avg/max/mdev = 0.071/0.802/3.177/1.203 ms
mininet> █
```



The output of the network once

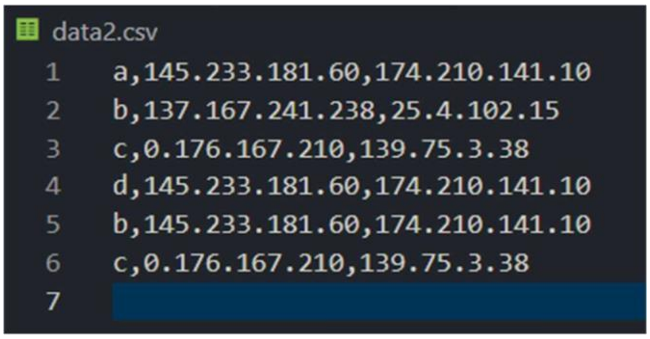
executed: Input csv file:



data.csv

```
a, 127.0.0.1, 192.168.2.5
b, 8.8.8.8, 126.89.34.2
c, 1.2.3.4, 192.168.2.4
d, 127.0.0.1, 192.168.2.5
b, 127.0.0.1, 192.168.2.5
c, 1.2.3.4, 192.168.2.4
```

Encrypted ip’s csv file:

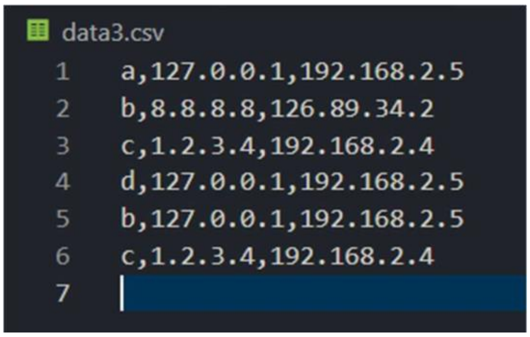


data2.csv

```
a,145.233.181.60,174.210.141.10
b,137.167.241.238,25.4.102.15
c,0.176.167.210,139.75.3.38
d,145.233.181.60,174.210.141.10
b,145.233.181.60,174.210.141.10
c,0.176.167.210,139.75.3.38

```

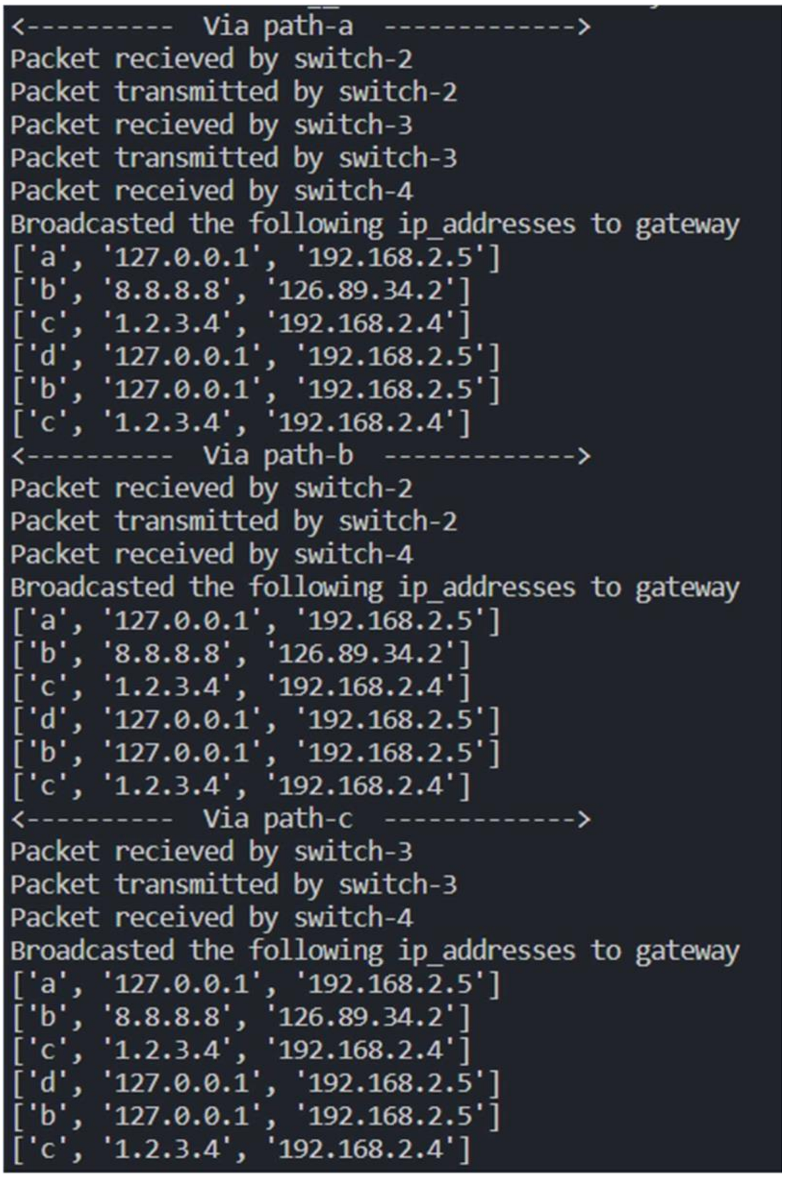
Decrypted ip’s csv file:



data3.csv

```
a,127.0.0.1,192.168.2.5
b,8.8.8.8,126.89.34.2
c,1.2.3.4,192.168.2.4
d,127.0.0.1,192.168.2.5
b,127.0.0.1,192.168.2.5
c,1.2.3.4,192.168.2.4

```



```
<----------  Via path-a  ------------->
Packet recieved by switch-2
Packet transmitted by switch-2
Packet recieved by switch-3
Packet transmitted by switch-3
Packet received by switch-4
Broadcasted the following ip_addresses to gateway
['a', '127.0.0.1', '192.168.2.5']
['b', '8.8.8.8', '126.89.34.2']
['c', '1.2.3.4', '192.168.2.4']
['d', '127.0.0.1', '192.168.2.5']
['b', '127.0.0.1', '192.168.2.5']
['c', '1.2.3.4', '192.168.2.4']
<----------  Via path-b  ------------->
Packet recieved by switch-2
Packet transmitted by switch-2
Packet received by switch-4
Broadcasted the following ip_addresses to gateway
['a', '127.0.0.1', '192.168.2.5']
['b', '8.8.8.8', '126.89.34.2']
['c', '1.2.3.4', '192.168.2.4']
['d', '127.0.0.1', '192.168.2.5']
['b', '127.0.0.1', '192.168.2.5']
['c', '1.2.3.4', '192.168.2.4']
<----------  Via path-c  ------------->
Packet recieved by switch-3
Packet transmitted by switch-3
Packet received by switch-4
Broadcasted the following ip_addresses to gateway
['a', '127.0.0.1', '192.168.2.5']
['b', '8.8.8.8', '126.89.34.2']
['c', '1.2.3.4', '192.168.2.4']
['d', '127.0.0.1', '192.168.2.5']
['b', '127.0.0.1', '192.168.2.5']
['c', '1.2.3.4', '192.168.2.4']
```

Future Scope:

The approach proposed has the potential to become a breakthrough in the field of security in SDN. Some of the applications which could be further incubated soon could be:

1. Extension of the system to identify whether the incoming IP is encrypted or not. Based on this, the switches within the network too could communicate and utilize this technology for securing intra-network communication.
2. Practically implementing the code within the SDN topology for effective analysis of the proposed scheme.
3. Extending the system to support full strength of networks in real time scenario.
4. Secure transmission and storage of key within the architecture for enhanced security.
5. Enhancing the algorithm to prevent cryptanalysis.

Conclusion:

In conclusion, our project has been dedicated to addressing a critical security concern within Software-Defined Networking (SDN) environments—namely, the need to protect IP address information from passive attacks. We introduced a novel approach by implementing encryption algorithms at the first switch in the network and decryption algorithms at the last switch, ensuring end-to-end IP address encryption to prevent data interception and eavesdropping.

Throughout the course of this project, we conducted extensive research, selected appropriate encryption and decryption algorithms, and integrated them into the SDN architecture. We thoroughly evaluated the impact of this security mechanism on network performance, considering factors such as latency, throughput, and resource utilization. Our findings indicate that our approach successfully enhances the security posture of SDN networks while maintaining network efficiency within acceptable bounds.

This project represents a significant step forward in the ongoing efforts to secure SDN environments. By safeguarding IP address data, we provide a valuable layer of defense against passive attacks, ultimately contributing to the confidentiality and integrity of sensitive information. Furthermore, our work serves as a testament to the adaptability and extensibility of SDN architectures, showcasing the potential for innovative solutions in addressing evolving security challenges.

In conclusion, our project's success in implementing end-to-end IP address encryption and decryption within SDN networks demonstrates our commitment to strengthening security in a dynamic and transformative networking paradigm. We hope our work serves as a foundation for further advancements in SDN security, ensuring the continued integrity of data in an interconnected world.

References:

Wallker, Peter, et al. "Anonymous network based on software defined networking." *2020 4th International Conference on Trends in Electronics and Informatics (ICOEI)(48184)*. IEEE, 2020.