Journal of Scientific & Industrial Research
Vol. 83, September 2024, pp. 977-988
DOI: 10.56042/jsir.v83i9.6313

An Authentication Mechanism to Prevent Various Security Threats in Software Defined Networking by using AVISPA

Anil Ram[1]*, Manash Pratim Dutta[2] & Swarnendu Kumar Chakraborty[1]*

[1]Department of Computer Science and Engineering, National Institute of Technology, Jote 791 113, Arunachal Pradesh, India
[2]Department of Computer Science & Information Technology, Cotton University, Pan Bazaar, Guwahati, 781 001, Assam, India

Received 21 October 2023; revised 13 June 2024; accepted 23 August 2024

Scalability in Software Defined Networking (SDN) empowers extensive interconnectivity among devices, making it particularly advantageous. As the number of hosts in SDN networks grows in response to increasing demand, network administrators must ensure the legitimacy of these hosts. To address this, our method requires SDN hosts to be authenticated before connecting to the SDN controller using the Kerberos authentication protocol. Kerberos employs a centralized server to validate host credentials, making it easier for hosts to access network rules and communicate securely with the controller. For enhanced security, we use Automated Validation of Internet Security Protocols and Applications (AVISPA), which automates the verification of security protocols, identifying vulnerabilities early and improving secure application development. AVISPA employs protocols like OFMC (Otway-Rees Formal Model of Communication) and CL-Atse (Computational Logic for Automated Security) for security checks, which are effective for our analysis. In the OFMC evaluation of our technique, 564 nodes were visited with a search time of 0.23 seconds and a depth of 10 plies, indicating favourable results for network security, data integrity, transparency, reliability, and confidentiality. The CL-Atse analysis examined 545 states, with 506 nodes reachable in 0.12 seconds, demonstrating security against Man-in-the-Middle (MIM) and Replay attacks. The computational cost was 0.0982 milliseconds, proving that our technique is secure against various threats while maintaining low computational overhead.

Keywords: Computational logic for automated security, Encryption, Kerberos authentication protocol, Otway-Rees formal model of communication, Traffic flow

Introduction

SDN has arisen to address the shortcomings of traditional network architectures by decoupling the Control Plane (CP) from the Data Plane (DP).[1–5] This separation enhances network flexibility and reliability significantly. In SDN, the CP takes on the role of overseeing the network[6–8], where it defines and manages rules and policies governing traffic flow and network behavior. Meanwhile, the DP executes these directives by forwarding packets and managing traffic based on the rules established by the CP.[6–11] The OpenFlow protocol plays a crucial role in facilitating communication between the CP and DP elements of the SDN architecture, allowing centralized management and control.[12,13] This centralized approach leverages a network operating system running on the SDN controller, which provides comprehensive oversight and management capabilities across the entire network infrastructure. By separating control functions from data forwarding tasks, SDN enables dynamic network management, efficient resource allocation, and easier implementation of network-wide policies and security measures.[14–16] This paradigm shift not only enhances network agility and scalability but also simplifies network administration and troubleshooting, making SDN a cornerstone of modern network architectures.

SDN brings a revolutionary level of programmability to network management, empowering administrators to rapidly deploy and adjust network policies. However, this flexibility also introduces complexities that heighten security challenges, especially in scaling SDN networks handling numerous devices.[17–21] Authentication emerges as a critical factor in ensuring secure data exchange among network components. Devices must authenticate themselves to establish trust and effectively manage access privileges within the SDN infrastructure.[22–28] Robust authentication mechanisms, such as digital certificates or strong authentication protocols like EAP-TLS, are essential to verify the identity of devices and prevent unauthorized access or malicious activities.[29,30] Secure authentication not only safeguards data integrity and

*Author for Correspondence
E-mail: anilram.nitap@gmail.com; swarnendu@nitap.ac.in

confidentiality but also reinforces the overall trustworthiness of the SDN environment. As SDN networks evolve and expand, maintaining stringent authentication practices becomes increasingly vital to mitigate risks associated with potential vulnerabilities and ensure the reliability and security of network operations.[23,25] By integrating robust authentication protocols, SDN administrators can effectively manage network access and uphold the integrity of communications across diverse and dynamic network environments.

Data security within a network relies on the exchange of secret keys for encryption and decryption between devices.[24,28,31] An authorized Authentication Server (AS) plays a pivotal role in this process by managing and distributing these keys across the network. The AS maintains a centralized database containing IDs of all network devices, ensuring that authentication requests originate from trusted sources.[26,29] It oversees the authentication process, verifying the identity of devices seeking access and ensuring secure communication channels throughout the network, whether between users and servers or among various network components. This centralized management not only enhances security but also streamlines the administration of authentication policies and key distribution, minimizing the risk of unauthorized access or data breaches. By securely managing and distributing encryption keys, the AS ensures that only authenticated devices can participate in secure data exchanges, thereby safeguarding sensitive information and maintaining the integrity of network communications.[24,26,29,32,33] This robust authentication framework is essential for establishing and maintaining a secure network environment where confidentiality, integrity, and availability of data are preserved against potential threats and vulnerabilities. The Authentication System (AUS)[24,27,30,31,33] requirement meets the following four requirements:

1) Security: - The AUSmust be durable enough to recognize and reject bogus requests for any service.
2) Reliability: - It's crucial that the authentication servers can backup one another.
3) Transparency: - If authentication involves more than just entering passwords, users won't be harmed.
4) Scalability: -The AUS must be scalable in order to handle any extra users or servers that are added to the network.

The aforementioned Authentication criteria were satisfied, so we decided to use the Kerberos authentication protocol[24,29,33] in the suggested technique's authentication protocol because it complies with all four criteria, which is discussed in the Proposed Technique section.

Kerberos Authentication Protocol

A trustworthy third-party authentication server is used by the Kerberos network authentication protocol to provide secure authentication for client-server applications. Here is how the Kerberos authentication technique operates:

The Kerberos Authentication Server (KAS) receives a request from a user trying to access a network resource (such as a file server). A Ticket-Granting Ticket (TGT), encrypted with the user's password, and is returned by the KAS to the user. The user requests a service ticket for the requested network resource and submits the TGT to the Ticket Granting Server (TGS). A service ticket for the requested resource is generated by the TGS once it has verified the TGT and is encrypted using a shared secret key between the TGS and the resource server. The resource server receives the user's service ticket, validates it, and then allows access to the requested resource. Kerberos uses mutual authentication and encryption between the client, the KAS, the TGS, and the resource server to offer strong authentication and security. Users get benefit from single sign-on since they have to authenticate only once to access a variety of network resources.[24,29,31,34,35]

The 6 steps of Kerberos authentication method is depicted in Fig. 1. (1) A user requests TGT to Key Distribution Center (KDC), Authentication Server (AS) verifies the request whether it is authenticated

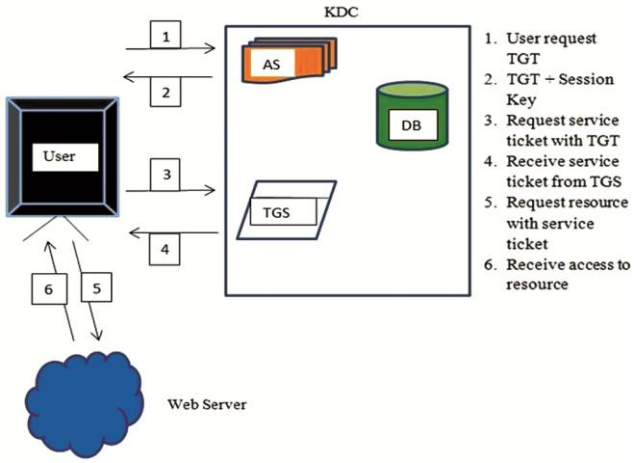

Fig. 1 — Kerberos authentication method

one or not; (2) AS forwarded to TGS for ticket generation, now the KDC sends TGT with session key to the user; (3) Later on, that a user sent request service ticket with TGT to KDC (i.e., AS, Database (DB), TGT) for authentication purpose; (4) After authentication the KDC sends receive service ticket from TGS to the authenticated user; (5) After handshake done by the user to KDC, now the user sends request resource with service ticket to Web server, (6) In the last part, the Web Server sends the receive access to resource to the authenticated user.

Related work Section provides an overview of current related research and developments. In Proposed technique Section, the paper presents the proposed technology along with its systematic approach. Simulation and Validation section rigorously examines the validation and replication processes of this approach, ensuring its reliability and applicability. Discussion Section concludes by discussing the strategy's robustness against various network attacks, highlighting its effectiveness in enhancing network security and resilience.

Related Work

One of the key concerns that network administrators and researchers have working to improve the effectiveness of SDN that is throughput, jitter, availability, RTT etc but the main concern area to look out is security over SDN so that network devices may trust one another, a security feature that requires improvement is authentication. The following are some of the research studies that have been conducted on this topic:

An authentication method named AUTH Flow was proposed by Diogo *et al.*, the vital concept behind this approach is to use layer 2 protocols to perform network-wide authentication between hosts and servers. In this suggested technique, the IEEE 802.1X standard is used. Authentication for the network's hosts is also provided by the RADIUS authentication server. In this approach, the messages sent between the hosts and the RADIUS servers are encapsulated using the Extensible Authentication Protocol (EAP). Additionally, a server known as an authenticator was established, whose job it is to convert IEEE 802.1X packets into RADIUS packets. Working at the application layer's SDN controller is AUTH Flow.[15]

The primary SDN attacks that could compromise the entire network were categorised and described by Hamza *et al.* also, mentioned were several potential defences against these assaults.[16] For the apps used in the SDN environment, Latif *et al.* developed an authentication system. Between the user request and the untrusted network, this technique implemented an authentication scheme. This is seen as one of the major difficulties with SDN networks. The Zero-Knowledge protocol serves as the foundation for this authentication system.[17] Aditya *et al.* defines security techniques to prevent only man-in-middle attack and Denial of Service (Dos) attack but due to the lack of the security mechanism in data layer there is much possibility to happen replay attack in their existing scheme.[18] Midha *et al.* discusses the privacy and authentication method for securing healthcare management system, confidentiality, reliability but some attacks like impersonation attacks, man-in-middle attack still vulnerable to their proposed method due to the lack of weak authentication technique was used.[19]

In Table 1, Diogo *et al.*[15] methods found that Dos attack and Spoofing attack were prevented but some attacks like replay attack, Man-in-Middle (MIM) attack still vulnerable to their proposed method. However, Hamza *et al.'s* methods, focuses to prevent MIM attack and DoS attack but some attack still haunting in traffic flow, secure communication.[16] Impersonation attack, replay attack, spoofing attack were prevented by the Latif *et al.'s*[17] scheme but MIM attack and DoS attack are still vulnerable to their

Table 1 — Comparative analysis of the author's work on security and authentication over security vulnerabilities (χ represent the meaning of "NO" and √ symbol represent "Yes")

Authors name	Replay attack	Impersonation attack	Man-in-Middle attack	Denial of Service attack	Spoofing attack
Diogo *et al.*[15]	χ	√	χ	√	√
Hamza *et al.*[16]	χ	χ	√	√	χ
Latif *et al.*[17]	√	√	√	χ	√
Aditya *et al.*[18]	χ	χ	√	√	√
Midha *et al.*[19]	χ	χ	χ	√	√
Savic *et al.*[20]	χ	√	√	χ	√
Karmakar *et al.*[21]	χ	√	√	χ	√
Elsayed *et al.*[22]	χ	√	√	√	χ
Wang *et al.*[23]	√	χ	√	√	χ

proposed scheme, Aditya *et al.*[18] design an authentication scheme which can prevent MIM attack, DoS attack and Spoofing attack but some security threats like replay attack, Impersonation attack still unable to handle. Midha *et al.*[19] focuses on to prevent DoS and spoofing attack for healthcare management system but some attack like replay attack, impersonation attack and MIM attack are still vulnerable to this proposed scheme.

Savic *et al.*[20] discussed the robustness of RSA encryption over decades of research, highlighting that while various attacks have been explored, none have proven catastrophic. Emphasis is placed on identifying and avoiding implementation pitfalls rather than flaws in RSA's core design. This perspective reinforces RSA's security under proper parameter selection, affirming its continued reliability in cryptographic applications. RSA's security can be affected by key size, leading to increased encryption and decryption durations, particularly with larger keys needed to resist contemporary computational capabilities. Theoretical progress in quantum computing, exemplified by algorithms like Shor's algorithm, poses a potential future risk to RSA. This is because such algorithms could enable practical prime factorization, which RSA depends on for security. RSA implementations may also be susceptible to side-channel attacks that exploit unintentionally leaked information, such as timing or power usage. The algorithm's security fundamentally rests on the challenge of factoring large prime numbers, implying that any advancement in factoring techniques could potentially weaken its defences.

Karmakar *et al.*[21] presents a comprehensive threat model for SDN, highlighting potential vulnerabilities and specifically focusing on SDN topology poisoning attacks. It emphasizes the importance of secure network application development and proposes a Security Application designed for SDN controllers and OpenFlow switches. This application implements robust security and intrusion detection policies, enforced by agents on switching hardware. The approach is validated using Open Network Operating System (ONOS), an SDN Controller, to develop the Security Application and test against threat scenarios. The article concludes with a demonstration of how the proposed solution mitigates attack scenarios and includes a performance analysis showcasing its effectiveness in securing SDN environments. While the Security Application presented shows promise in securing SDN environments against various threats, there are several potential drawbacks and considerations: complexity, scalability, resource overhead, dependency on ONOS, and integration complexity. Addressing these drawbacks would be crucial for ensuring the practical deployment and long-term effectiveness of the proposed Security Application in real-world SDN environments.

Elsayed *et al.*[22] presents a detailed study focused on classical Machine Learning (ML) techniques applied to attack detection in SDN. It benchmarks these techniques using the NSL-KDD dataset and highlights their limitations in achieving high performance. The study identifies shortcomings in traditional ML methods when applied to the complex and dynamic environment of SDN, leading to suboptimal detection rates. The authors emphasize the need for more effective approaches and announce their plans to develop a deep learning (DL) - based framework. This framework is anticipated to surpass existing state-of-the-art methods in terms of performance and accuracy in detecting attacks within SDN environments. Several potential security vulnerabilities haunted this article such as feature engineering complexity, scalability issues, dependency in dataset quality, complexity of DL approaches, adaptability and real-time constraints, and performance variability. It is crucial to tackle these limitations to enhance the efficacy and practical utility of ML techniques.

Wang *et al.*[23] designed a network Intrusion Detection system (IDS) using a hybrid neural network that integrates flexible transformers and a Convolutional Neural Network (CNN) algorithm. The model they proposed achieved an impressive accuracy of 98.90%. However, integrating multiple models typically increases the complexity of the system, demanding more memory and CPU processing power. The proposed model extracts numerous characteristics, demanding increased memory and longer processing times. This could potentially bottleneck the controller and restrict its application in real-time usage within large and complex networks. Many of these extracted traits are irrelevant to the workings of network attacks. In Table 2 shows the software metrics where authors used in their research work.

Proposed Technique

Authentication is a safekeeping feature that wants to be enhanced by mapping the identity of the host to the Authentication Approval Server (AAS); this study's main goal is to establish a clear authentication

Table 2 — Software metrics used by authors

By/For SDN	Author	Techniques used	Security types	Targeted planes	Targeted interfaces
For	Diogo *et al.*[15]	Layer 2, IEEE 802.1x	Enhancement	Application plane	Northbound
For	Hamza *et al.*[16]	Kerberos Authentication Protocol	Framework	DP, CP	Southbound
By	Latif *et al.*[17]	Zero-knowledge	Enhancement	DP, CP	Southbound
By	Aditya *et al.*[18]	Support Vector Machine	Framework	DP	Southbound
For	Midha *et al.*[19]	Defensive Algorithm	Framework	CP	Southbound
By	Savic *et al.*[20]	RSA, Shor's Algorithm	Enhancement	DP, CP	Northbound, Southbound
By	Karmakar *et al.*[21]	ONOS, Raspbian Virtual Machine	Enhancement	CP	Southbound
By	Elsayed *et al.*[22]	NSL-KDD, Deep-Learning framework	Framework	CP	Northbound, Southbound
For	Wang *et al.*[23]	IDS, CNN algorithm	Enhancement	CP	Northbound, Southbound

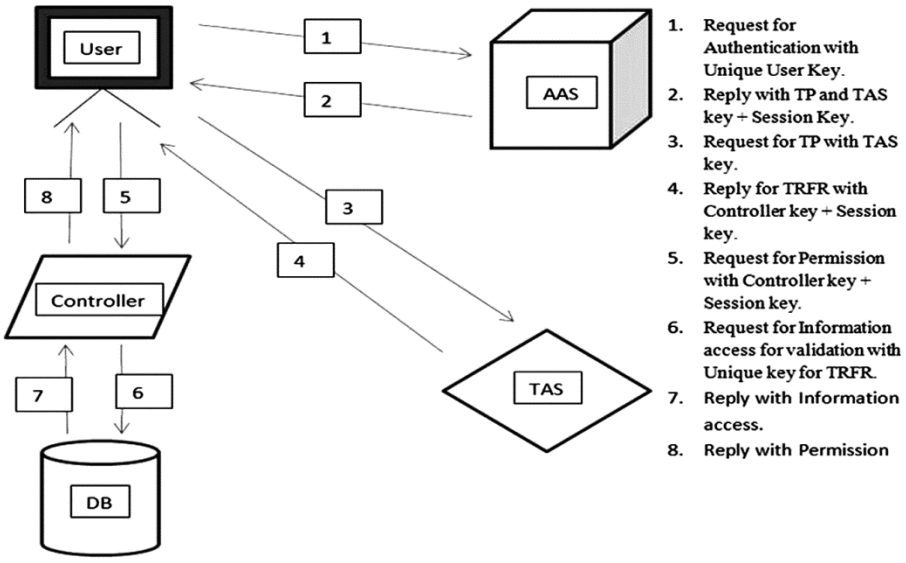

Fig. 2 — Proposed architecture

method between SDN network users and controllers allowing the user to ask the controller for network rules. In order for the user to request network instructions from the controller, building a clear authentication method between users and controllers by plotting the user's characteristics to the AAS is the primary goal of this paper. The suggested approach leverages the Kerberos authentication protocol to ensure robust and secure access to the network's services. The Kerberos authentication process in the SDN scenario will be managed by the AAS. In accordance with the behaviour of the request, the authentication server approves or rejects message requests to the controller. The request can communicate with the controller if the necessary authentication parameters are included. The request will be declined in any other cases. To obtain a facility ticket from Ticket Allowances Server (TAS), the user is given a Ticket Provider (TP) by the authentication server. The TAS must validate the user service access appeal before approving a Request for Rules (RFR). To start a dialogue between the two; the host could send the controller a request for the rule. All information about hosts and servers, including their identities, is stored on the authentication server. The architecture of the suggested method is depicted in Fig. 2 and the description of the notation used in Figs 2 & 3 is provided in Table 3.

In Fig. 2, the following steps of the proposed architecture are discussed:

1 Step 1: User Authentication Request — User sends a request to Authentication and Authorization Server (AAS) with a unique user key for identity authentication.

2 Step 2: AAS Response with Keys — AAS verifies the user's identity; AAS sends a reply to

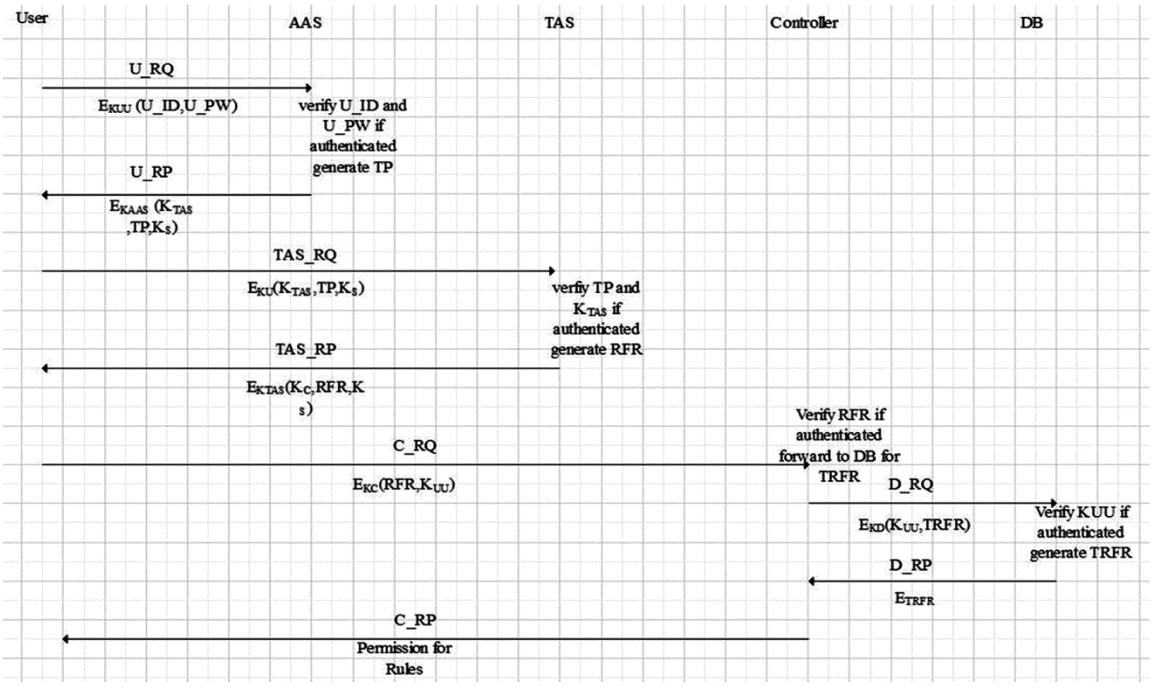

Fig. 3 — Methodical process of proposed authentication algorithm

Table 3 — Notation of the proposed architecture design

Notation	Description
AAS	Authentication approval server
TAS_RQ	Ticket-allowances server request
TAS	Ticket-allowances server
C_RQ	Controller request
TP	Ticket provider
K_{AAS}	Authentication approval server key
DB	Database
U_RQ	User request
K_C	Controller's key
U_RP	User reply
RFR	Request for rules
TAS_RP	Ticket-allowances server reply
C_RP	Controller reply
U_ID	User ID
U_PW	User password
K_S	Session key
K_{UU}	User unique key
K_{TAS}	Ticket allowances server key
E	Encryption
TRFR	Traffic request for rules

the user containing Traffic Processor (TP) and Traffic Authorization Service (TAS) keys along with a session key.

3 Step 3: TP Request to TAS — User sends a request to TAS with the TAS key to access the Traffic Processor (TP)

4 Step 4: TAS Verification and RTFR Response — TAS verifies the authenticity of the user; TAS replies to the user with a Response Traffic Flow Regulation (RTFR) message containing the controller key used for traffic flow to the destination address.

5 Step 5: Permission Request to Controller — User requests permission from the Controller; User includes the session key received from AAS and the controller key received from TAS in the request to the Controller.

6 Step 6: Controller Request to DB — Controller forwards the user request to the Database (DB) for information access validation; Controller includes the unique key for validation purposes.

7 Step 7: DB Verification and Reply — DB verifies the authenticated user and the request; DB sends a reply to the Controller with information access.

8 Step 8: Controller Permission Response to User — Controller sends a reply to the user granting permission based on the DB's response.

After verification and authentication process a user can send and receive information with the aid of controller. Applying session key, unique user key into the process of authentication makes the proposed procedure secure and efficient.

Results and Discussion

With AVISPA, a widely used tool for security protocol verification. AVISPA is an effective tool for automating security measures and cryptographic rules. There are four backends that offer the state-of-the-art AVISPA tool. Since CL-AtSe and OFMC support the suggested method, we have employed those two backend. Employing CL-AtSe and OFMC as backend tools holds significant importance for ensuring robust authentication mechanisms and preventing various security threats. CL-AtSe (Computational Logic for Automated Security) offers a methodical approach to confirming the accuracy of authentication protocols deployed in SDN controllers and switches.[24] This process aids in detecting and addressing security risks such as unauthorized access attempts or identity spoofing. These vulnerabilities are particularly significant in SDN setups where centralized controllers oversee network operations. OFMC (Otway-Rees Formal Model of Communication)[28] ensures that authentication protocols maintain confidentiality, integrity, and authentication characteristics. Through symbolic model checking methods, OFMC thoroughly examines potential states and interactions of protocols, effectively identifying and thwarting security risks such as man-in-the-middle attacks or unauthorized manipulation of data. By integrating CL-AtSe and OFMC as backend tools within the AVISPA framework, the paper enhances the overall security posture of SDN architectures. It enables rigorous validation of authentication mechanisms against a range of security threats, fostering trust and reliability in SDN deployments. Ultimately, the adoption of CL-AtSe and OFMC contributes to building resilient SDN infrastructures that can withstand evolving security challenges effectively. The codes for the suggested technique were written using the High-Level Protocol Specification Language (HPLSL). The Delov-Yao intrusion model is used by the HLPSL that can be used to launch MIM attacks and reply assaults to test the proposed protocol's security[30] as it is being executed. The simulation outcome will be non-violent if the suggested protocol is secure, else unsafe.

In Fig. 3, a detailed step-by-step process is illustrated, outlining the sequence from user request through verification, authentication, and concluding with the response, systematically explained across algorithms 1 to 4.

Algorithm 1: User authentication at AAS

```
Function UserAuthentication(E_UID, E_Pwd):
 // Step 1: User initiates access request with encrypted credentials
 // E_UID: Encrypted User ID
 // E_Pwd: Encrypted Password
 if AF(E_UID, E_Pwd) == True:
 // Step 3: AAS verifies credentials using AF
 TP = Encrypt(TAS, GenerateTokenPayload(), K_s)
 return TP
 else:
 return "Authentication Failed"
```

Algorithm 2: TAS verification and response

```
Function Process TASRequest (K_TAS, TP, K_s):
 // Step 4: User sends TAS request with K_TAS, TP, K_s
 // K_TAS: Key for TAS
 // TP: Token Payload
 // K_s: Session Key
 TAS, TP_decrypted, K_s_decrypted = Decrypt (TP, KTAS)
 if TAS == ValidTAS and K_s == K_s_decrypted:
 // Step 5: TAS verifies TP and generates RFR
 K_c = Generate Controller Key()
 RFR = Encrypt(K_c, K_s)
 return RFR
 else:
 return "Invalid Request"
```

Algorithm 3: Controller authentication and data request

```
Function Process Controller Request(RFR, K_UU):
 // Step 6: User sends access request to controller with RFR, K_UUK_UU):
 // RFR: Encrypted session key
 // K_UU: User Key
 K_c = Decrypt(RFR, ControllerKey)
 if AF(K_UU) == True:
 // Step 7: Controller verifies RFR and forwards request to DB
 TRFR = RetrieveDataFromDB()
 return TRFR
 else:
 return "Authentication Failed"
```

Algorithm 4: DB processing and response

```
Function Process DBRequest(TRFR, K_UU):
 // Step 8: DB processes request and sends encrypted TRFR to controller
 // TRFR: Encrypted trusted data
 // K_UU: User Key
 TrustedData = Decrypt(TRFR, K_UU)
 return TrustedData
```

In the simulation of the suggested technique, we have listed eleven safekeeping goals, five are to ensure the veracity of the communication strictures, and the remaining six of which are to certify the confidentiality of the service parameter. The suggested technique's HLPSL code includes these eleven security goals, and the simulation's outcome is based on them. Security aims are detailed in Table 4, providing an explanation of the goals related to safeguarding the system. We have used the OFMC

Table 4 — Security objectives of the Suggested Technique

Goal	Description
Secy_of_the_K_C	Controller Key secrecy
Secy_of _the_U_PW	Secrecy of the Unique User Password
Secy_of_the_RULE	Rules which are sent by the Controllers secrecy
Secy_of_the_TP	Secrecy of the Ticket Provider
Secy_of_the_K_S	Secrecy of the Session Key
Secy_of_the_RFR	Request for Rules secrecy
Secy_of_the_K_{TAS}	Secrecy of Ticket Allowances Server Key
Secy_of_the_TRFR	Secrecy of the Traffic Request for Rules
Auth_on_TP	Authenticity on Ticket Provider
Auth_on_K_{TAS}	Authenticity on Ticket-Allowances Ticket key
Auth_on RFR	Authenticity of Request for Rule
Auth_on K_C	Authenticity of the Controller key
Auth_on K_S	Authenticity of Session key
Auth_on RULE	Rules engaged by the Controllers authenticity
Auth_on TRFR	Authenticity of Traffic Request for Rule

Table 5 — The Proposed simulation result of OFMC back-end

```
  %OFMC
%Version of 2006/02/13
SUMMARY
SAFE
DETAILS
  BOUNDED_NUMBER_OF_SESSIONS
PROTOCOL
  /home/span/span/testsuite/results/customalgo.if
GOAL
  as_specified
COMMENTS
STATISTICS
  parseTime: 0.00s
  searchTime: 0.23s
 visitedNodes: 564 nodes
 depth: 10 plies
```

and CL-AtSe back-ends to run our code in accordance with these objectives.

The simulation output from the OFMC backend, as showed in the Table 5, confirms the security of the proposed solution against DoS, Impersonation, MIM, and Replay attacks. This indicates that comprehensive measures have been implemented to safeguard the system from these potential threats. To protect against DoS attacks, the system likely incorporates robust defences such as rate limiting and request validation mechanisms to mitigate excessive traffic or malicious requests. Effective measures against impersonation suggest the implementation of strong authentication protocols and secure authorization mechanisms, ensuring only legitimate users can access resources. Security against MIM attacks is typically achieved

Table 6 — The Proposed simulation result of CL-Atse back-end.

```
SUMMARY
SAFE
DETAILS
BOUNDED_NUMBER_OF_SESSIONS
TYPED_MODEL
PROTOCOL
/home/span/span/testsuite/results/customalgo.if
GOAL
As_specified
BACKEND
CL-Atse
STATISTICS
Analysed    : 0.00s
Reachable   : 0.23s
Translation : 564 nodes
Computation : 10 plies
```

through encryption protocols, which encrypt communications to prevent interception and tampering by unauthorized entities. Additionally, protection against replay attacks involves mechanisms such as nonce verification to ensure that each communication session is unique and not susceptible to replay by malicious actors. The positive outcome of the OFMC simulation underscores the system's resilience and adherence to good practices in network security, emphasizing its capability to maintain data integrity, reliability, transparency, confidentiality, and operational continuity in the face of potential network security threats. The simulation output from the Cl-Atse backend, showed in the Table 6, validates the resistance of the proposed work against MIM and Replay attacks. This signifies robust security measures integrated into the system architecture. Protection against MIM attacks suggests the implementation of secure communication protocols, which encrypt data to prevent interception and alteration by unauthorized intermediaries. Additionally, resilience against replay attacks indicates the utilization of methods like nonce verification to ensure that each transaction or communication session is unique and not vulnerable to being maliciously reused. The positive results from Cl-Atse highlight the effectiveness of these security mechanisms in safeguarding data integrity and confidentiality within the system. By addressing these specific threats, the solution ensures that communication channels remain secure and trustworthy, maintaining the integrity of exchanged information and safeguarding against potential vulnerabilities exploited by sophisticated attackers. This robust security posture not only protects

sensitive data but also enhances the reliability, scalability, and resilience of the system. The two simulations of the domino effect demonstrate that the suggested approach is effective, secure, and can enhance the safety of the SDN controller.

Low cost computation time is necessary for the SDN network to flow information smoothly within fraction of milliseconds, Our proposed computation cost time is mentioned in the Table 7, defines low computation cost time as compared to existing computation time. Our proposed computation cost time is explained briefly and in the previous research done by the researchers and our proposed technique with respect to computation is mentioned in the Table 8:

The proposed authentication system has undergone a comparison investigation with a number of other schemes, and we have discovered that it requires less processing complexity and time. According to what was previously indicated in the examination of the suggested mechanism, the user consumed 2TU and the controller produced 4TU + 2TC, consequently, the proposed mechanism's overall computation cost equals 6TU + 2TC.

In the following Table 9, many researchers implemented authenticated algorithm to secure SDN from vulnerability but the complexity of the separation of the DP from the CP makes researchers task difficult. To prevent the security threats to SDN architectures, researchers prevented the following security goals:

The proposed security method for protecting network hosts from attacks, such as user impersonation, controller impersonation, and Denial-of-Service (DoS) attacks, is designed to ensure data integrity and privacy in a network environment. Here's a detailed discussion of how each component of the suggested method addresses these attacks, along with insights from related studies:

User Impersonation Attack

The method addresses user impersonation by having hosts authenticate themselves to an Authentication and Authorization Server (AAS) using a set of credentials (ID, PW, and KUU). The AAS verifies these credentials before issuing a Token of Permission (TP) that allows the host to request network rules from the controller.

This approach is effective because it leverages a centralized authentication mechanism to ensure that only legitimate hosts can interact with the network controller. By validating each host's identity through AAS, the system prevents unauthorized entities from gaining access to the network resources or manipulating the network rules.

- Secure SDN Architectures: Studies such as "Fang *et al.*[7], Salam *et al.*[35], Sahana *et al.*[33], Midha *et al.*[19]", discuss similar mechanisms for validating host identities in SDN environments to prevent unauthorized access and manipulation.

- Authentication Protocols: Research in "Abbas *et al.*[34], Latif *et al.*[17], Wang *et al.*[23], Kayathri *et al.*[13], Aditya *et al.*[18]", highlights the importance of strong authentication mechanisms in preventing impersonation attacks, which aligns with the proposed method's approach.

Controller Impersonation Attack

The method prevents controller impersonation attacks by ensuring that the controller's identity and secret key (KC) are stored securely in the AAS database. Hosts receive a Token of Permission (TP),

Table 7 — Computation cost time of the proposed scheme

Participants	User	Controller
Computation complexity	2TU	4TU + 2TC
Computation Time	2*0.012=0.024 ms	4*0.0093=0.0372 2*0.0185= 0.0370
Total computation time	0.024 + 0.0742= 0.0982 ms	0.0372+0.0370= 0.0742 ms

Note: TU= Time taken of User, TC= Time taken by Controller

Table 8 — Comparative analysis of the proposed technique with respect to computation cost

Authors	Computational complexity	Computation time (in millisecond)
Diogo *et al.*[15]	10TU+2TC	0.1015
Hamza *et al.*[16]	9TU+3TC	0.0993
Latif *et al.*[17]	34TU+2TC	0.3910
Aditya *et al.*[18]	4TU+3TC	0.1253
Midha *et al.*[19]	2TU+2TC	0.0989
Abbas *et al.*[34]	8TU+2TC	0.1011
Fang *et al.*[7]	7TU+2TC	0.1106
Wang *et al.*[23]	3TU+4TC	0.0998
Soares *et al.*[9]	10TU+8TC	0.2534
Sahana *et al.*[33]	3TU+3TC	0.0989
Irshad *et al.*[11]	16TU	0.0125
Jagtap *et al.*[31]	9TU+3TC	0.1178
Kayathri *et al.*[13]	2TU+5TC	0.1023
Salam *et al.*[35]	2TU+5TC	0.0990
Savic *et al.*[20]	3TU+3TC	0.1068
Karmakar *et al.*[21]	8TU+2TC	0.1276
Elsayed *et al.*[22]	2TU+2TC	0.1128
Proposed technique	6TU+2TC	0.0982

Table 9 — Comparison the proposed technique between the works done by the researchers

Authors	MIM attack	DoS attack	Impersonation attack	PS attack	DP attack	CP attack	AP attack
Diogo *et al.*[15]	✗	√	√	√	✗	√	✗
Hamza *et al.*[16]	√	√	✗	✗	√	✗	✗
Latif *et al.*[17]	✗	✗	√	√	✗	√	✗
Aditya *et al.*[18]	√	√	✗	✗	✗	√	√
Midha *et al.*[19]	✗	√	✗	√	√	✗	✗
Abbas *et al.*[34]	✗	√	√	√	✗	√	√
Fang *et al.*[7]	✗	✗	√	√	√	✗	✗
Wang *et al.*[23]	✗	√	√	✗	✗	√	√
Soares *et al.*[9]	✗	✗	√	√	✗	√	√
Sahana *et al.*[33]	✗	√	✗	√	✗	√	✗
Irshad *et al.*[11]	✗	√	√	✗	✗	√	✗
Jagtap *et al.*[31]	✗	✗	√	√	√	✗	✗
Kayathri *et al.*[13]	✗	✗	√	√	✗	√	✗
Salam *et al.*[35]	✗	✗	√	√	✗	√	✗
Savic *et al.*[20]	✗	√	√	√	✗	√	✗
Karmakar *et al.*[21]	√	✗	√	√	✗	√	√
Elsayed *et al.*[22]	√	√	√	√	✗	√	✗
Proposed technique	√	√	√	√	√	√	√

Note: MIM: Man-in Middle attack, DoS: Denial of Service attack, PS: Packet Sniffing, DP: Data Plane, CP: Control Plane, AP: Application Plane.

which includes this key and other identifiers (K_{TAS} and RFR). This makes it difficult for an attacker to masquerade as the controller and issue malicious rules.

By incorporating the controller's secret key into the authentication process and validating it against the AAS database, the system ensures that only the genuine controller can issue rules to the hosts. This approach mitigates the risk of attackers injecting harmful rules or disrupting network behaviour.

- Controller Security in SDN: Research such as "Karmakar *et al.*[21], Diogo *et al.*[15], Hamza *et al.*[16]", highlights various strategies to secure SDN controllers, including the use of cryptographic techniques to prevent controller impersonation.

- Secure Key Management: The study "Midha *et al.*[19], Fang *et al.*[7], Kayathri *et al.*[13]", explores effective key management practices that enhance security against controller impersonation.

Denial-of-Service (DoS) Attack

To counter DoS attacks, the proposed method ensures that the controller only processes requests that are correctly formatted and encrypted. Specifically, the method checks that the Controller Request (C_RQ) contains the expected Response Format (RFR) and is encrypted using KC.

This approach protects the controller from being overwhelmed by malicious or malformed requests. By validating the format and encryption of incoming requests, the system can filter out illegitimate requests and prevent them from reaching the controller, thus maintaining network functionality.

- DoS Attack Mitigation in SDN: Research like "Soares *et al.*[9], Abbas *et al.*[34] Jagtap *et al.*[31]", provides insights into various techniques for protecting network controllers from DoS attacks, including request validation and rate limiting.

- Network Defense Mechanisms: The paper "Irshad *et al.*[11], Salam *et al.*[35], Sahana *et al.*[33]", discusses strategies for defending network infrastructure against DoS attacks, which are relevant to the proposed method's approach.

The proposed security method effectively addresses the identified network attacks by implementing robust authentication and authorization mechanisms, ensuring the integrity of communication between hosts and controllers, and protecting against DoS attacks through validation and encryption. The integration of these mechanisms is supported by related studies, which emphasize the importance of secure authentication, key management, and request validation in safeguarding network environments.

Conclusions

The study explores the implementation of Kerberos authentication within SDN environments to enhance network security and streamline the authentication

process for hosts connecting to the controller. By leveraging the Kerberos protocol, an authentication server verifies the identities of hosts, thereby ensuring the authenticity and integrity of network communications. This method addresses the critical need for robust security measures in SDN settings where numerous users connect dynamically, minimizing the risk of network failures and unauthorized access. However, the study acknowledges certain limitations, such as potential overhead in managing the authentication server and the complexity of integrating Kerberos with SDN architectures. Future research could delve deeper into optimizing authentication mechanisms between SDN controllers and switches. Furthermore, there is a notable opportunity to strengthen northbound APIs to facilitate secure integration of SDN controllers with external applications, thereby extending the scope of network management and security protocols. This research underscores the significance of advanced authentication solutions in securing SDN environments against evolving various security threats.

References

1 López-Millán G, Marín-López R, Pereñíguez-García F, Canovas O & Espín J A P, Analysis and practical validation of a standard SDN-based framework for IPsec management, *Comput Stand Interfaces*, **83** (2023) 103665, https://doi.org/10.1016/j.csi.2022.103665.

2 Aladaileh M A, Anbar M, Hintaw A J, Hasbullah I H, Bahashwan A A, Al-Amiedy T A & Ibrahim D R, Effectiveness of an entropy-based approach for detecting low-and high-rate DDoS attacks against the SDN controller: Experimental analysis, *Appl Sci*, **13(2)** (2023) 775, https://doi.org/10.3390/app13020775.

3 Gocher H, Taterh S & Dadheech P, Impact analysis to detect and mitigate distributed denial of service attacks with Ryu-SDN Controller: A comparative analysis of four different machine learning classification algorithms, *SN Comput Sci*, **4(5)** (2023) 456, https://doi.org/10.1007/s42979-023-01842-w.

4 Bhattacharya A, Rana R, Datta S & Venkanna U, P4-sknock: A two level host authentication and access control mechanism in p4 based sdn, *27th IEEE APCC*, (2022, October) 278–283.

5 Tuan D T, Duy P T & Pham V H, A blockchain-based authentication and access control for smart devices in sdn-enabled networks for metaverse, *9th IEEE NAFOSTED Conf Inf Comput Sci*, (2022, October) 123–128, https://doi.org/10.1109/NICS56915.2022.10013416.

6 Ram A, Dutta M P & Chakraborty S K, A flow-based performance evaluation on RYU SDN controller, *J Inst Eng India Ser B*, **105(2)** 203–215, https://doi.org/10.1007/s40031-023-00982-0.

7 Fang L, Li Y, Yun X, Wen Z, Ji S, Meng W & Tanveer M, THP: A novel authentication scheme to prevent multiple attacks in SDN-based IoT network, *IEEE Internet Things J*, **7(7)** (2019) 5745–5759.

8 Ahmed N, Ngadi A B, Sharif J M, Hussain S, Uddin M, Rathore M S & Zuhra F T, Network threat detection using machine/deep learning in sdn-based platforms: A comprehensive analysis of state-of-the-art solutions, discussion, challenges, and future research direction, *Sensors*, **22(20)** (2022) 7896.

9 Soares A A, Lopes Y, Passos D, Fernandes N C & Muchaluat-Saade D C, 3AS: Authentication, authorization, and accountability for sdn-based smart grids, *IEEE Access*, **9** (2021) 88621–88640.

10 Abdou A, Van O P C & Wan T, Comparative analysis of control plane security of SDN and conventional networks, *IEEE Commun Surv Tutor*, **20(4)** (2018) 3542–3559, https://doi.org/10.1109/COMST.2018.2839348.

11 Irshad A, Mallah G A, Bilal M, Chaudhry S A, Shafiq M & Song H, SUSIC: A secure user access control mechanism for SDN-enabled IIoT and cyber–physical systems, *IEEE Internet Things J*, **10(18)** (2023) 16504–16515, https://doi.org/10.1109/JIOT.2023.3268474.

12 Midha S, Novel approach to detect anomalies using defensive algorithm in SDN flows, *Turk J Comput Math Educ*, **12(2)** (2021) 536–542.

13 Kayathri T L, Kumaresan N & Vijayabhasker R, SDBGPChain: A decentralized low complexity framework to detect and prevent the BGP attacks using SDN with smart contract based dendrimer tree blockchain, *Computer Networks*, **230** (2023) 109800, https://doi.org/10.1016/j.comnet.2023.109800.

14 Ram A & Chakraborty S K, Analysis of Software-Defined Networking (SDN) Performance in wired and wireless networks across various topologies, including single, linear, and tree structures, *Indian J Inf Sources Serv*, **14(1)** (2024) 39–50, https://doi.org/10.51983/ijiss-2024.14.1.3926.

15 Pontes D F T, Caetano M F, Rocha F G P, Granville L Z & Marotta M A, On the transition of legacy networks to SDN-an analysis on the impact of deployment time, number, and location of controllers, *IFIP/IEEE Int Symp Integr Netw Manag*, (2021) 367–375.

16 Hamza A, Gharakheili H H, Benson T A & Sivaraman V, Detecting volumetric attacks on lot devices via sdn-based monitoring of mud activity, *Proc 2019 ACM Symp SDN Res* (2019) 36–48.

17 Latif Z, Lee C, Sharif K & Helal S, SDBlockEdge: SDN-blockchain enabled multihop task offloading in collaborative edge computing, *IEEE Sensors J*, **22(15)** (2022) 15537–15548.

18 Sumadi F D S & Aditya C S K, Comparative analysis of DDoS detection techniques based on machine learning in openflow network. *3rd IEEE Int Sem Res Inf Technol Intell Syst*, (2020) 152-157, https://doi.org/10.1109/ISRITI51436.2020.9315510.

19 Midha S & Tripathi K, Extended security in heterogeneous distributed SDN architecture, *Int Conf Adv Commun Comput Technol*, (2019) (991–1002), https://doi.org/10.1007/978-981-15-5341-7_75.

20 Savić D, Milić P, MAŻINJANIN B & Spalević P, Cryptanalytic attacks on RSA algorithm and its variants, *Prz Elektrotech*, **98(2)** (2022).

21 Karmakar K K, Varadharajan V & Tupakula U, Mitigating attacks in software defined networks, *Cluster Comput*, **22**

(2019) 1143–1157, https://doi.org/10.1007/s10586-018-02900-2.

22 Elsayed M S, Le-Khac N A, Dev S & Jurcut A D, Machine-learning techniques for detecting attacks in SDN, *IEEE 7th Int Conf Comput Sci Network Technol*, (2019) 277–281.

23 Wang C, Zhang Y, Chen X, Liang K & Wang Z, SDN-based handover authentication scheme for mobile edge computing in cyber-physical systems, *IEEE Internet Things J*, **6(5)** (2019) 8692–8701.

24 Ran L, Cui Y, Guo C, Qian Q, Shen G & Xing H, Defending saturation attacks on SDN controller: A confusable instance analysis-based algorithm, *Computer Networks*, **213** (2022) 109098.

25 Iqbal W, Abbas H, Deng P, Wan J, Rauf B, Abbas Y & Rashid I, ALAM: Anonymous lightweight authentication mechanism for SDN-enabled smart homes, *IEEE Internet Things J*, **8(12)** (2020) 9622–9633.

26 Usman M, Amin R, Aldabbas H & Alouffi B, Lightweight challenge-response authentication in SDN-based UAVs using elliptic curve cryptography, *Electronics*, **11(7)** (2022) 1026.

27 Ke C, Zhu Z, Xiao F, Huang Z & Meng Y, SDN-based privacy and functional authentication scheme for fog nodes of smart healthcare, *IEEE Internet Things J*, **9(18)** (2022) 17989–18001.

28 Manso C, Vilalta R, Muñoz R, Yoshikane N, Casellas R, Martínez R & Morita I, Scalability analysis of machine learning QoT estimators for a cloud-native SDN controller on a WDM over SDM network, *J Opt Commun Netw*, **14(4)** (2022) 257–266, https://doi.org/10.1364/JOCN.449009.

29 Singh A, Kaur N & Kaur H, Extensive performance analysis of OpenDayLight (ODL) and open network operating system (ONOS) SDN controllers, *Microprocess Microsyst*, **95** (2022) 104715.

30 Rajaram S, Vollala S & Ramasubramanian N, ERMAP: ECC-based robust mutual authentication protocol for smart grid communication with AVISPA simulations, *Int J Ad Hoc Ubiquitous Comput*, **41(4)** (2022) 232–245, https://doi.org/10.1504/IJAHUC.2022.126783.

31 Jagtap R R & Paradeshi S A, CRC method in SDN networks, *AIP Conf Proc*, **2717(1)** (2023), https://doi.org/10.1063/5.0143433.

32 Zhu L, Karim M M, Sharif K, Xu C, Li F, Du X & Guizani M, SDN controllers: A comprehensive analysis and performance evaluation study, *ACM Comput Surv*, **53(6)** (2020) 1–40, https://doi.org/10.1145/3421764.

33 Sahana D S & Brahmananda S H, Secure authentication framework for sdn-iot network using keccak-256 and bliss-b algorithms, *Int J Inf Technol*, **15(1)** (2023) 335–344, https://doi.org/10.1007/s41870-022-01074-w.

34 Yazdinejad A, Parizi R M, Dehghantanha A & Choo K K R, Blockchain-enabled authentication handover with efficient privacy protection in SDN-based 5G networks, *IEEE Trans Netw Sci Eng*, **8(2)** (2019) 1120–1132, https://doi.org/10.1109/TNSE.2019.2937481.

35 Salam R, Roy P K & Bhattacharya A, DC-IIoT: A secure and efficient authentication protocol for industrial internet-of-things based on distributed control plane, *Internet Things*, **22** (2023) 100782.