A Secured OpenFlow Protocol Using Elliptic Curves Cryptographic for Software Defined Networks

Layth A Khalil Al Dulaimi¹, R Badlishah Ahmad², Naimah Yaakob¹ and Qasim Mohammed Hussein³

¹ Embedded Networking and Advance Computing Cluster, School of Computer & Communication Engineering, Universiti Malaysia Perlis (UniMAP) Tingkat 1, Kampus Tetap Pauh, Putra 02600 Arau, Perlis.
² Faculty of Informatics and Computing, Universiti Sultan Zainal Abidin, 21300 Kuala Terengganu, Terengganu, Malaysia.
³ Control and Computer Engineering Department, Tikrit University, Tikrit - Iraq
laythadnan@studentmail.unimap.edu.my

Abstract. A Software-Defined Network (SDN) architecture was proposed to enhance the performance, flexibility and scalability of networks. However, SDN features such as centralized controlling, network programmability and virtualization present new security challenges to networks. This article will address security challenges related to SDN connection data transfer channels (i.e., control layer and infrastructure layer) and propose a novel connection data transfer channels architecture based on an OpenFlow protocol. Existing SDN OpenFlow protocols rely on higher layer secure mechanisms such as TLS (Transport Layer Security)/ SSL (Secure Sockets Layer) sessions. It is concluded that the transfer of secured data using elliptical curve encryption provides a more efficient SDN network. The proposed architecture was implemented in a testbed and its security features were analyzed.

1. Introduction
Secure Software-Defined Network data transfer between the control layer and infrastructure layer using elliptic curve cryptography is based on encryption and decryption, the proposed security mechanisms on both layers. The SDN architecture comprises three planes: Application layer, Control layer and Infrastructure layer. (APIs) Open Application Programmable Interfaces communicate between the Control and Infrastructure layers.

• Application Layer: The application layer consists of the end-user business applications and other control elements. The limit between the application layers and control layers is transferred by the northbound Open Application Programmable Interfaces (API). The SDN architecture offers different benefits such as control, flexibility, effective fragmentation, network management, network control, low operating cost, low cost plug-in devices, upon request and the provision of resources [1].

• Control Layer: The control layer consists of a centralized controller which provides unified oversight functions. Essentially, the controller supervises the packet forwarding functions of the network through an interface. Additionally, it controls all the network processes such as routing, indoctrination,
finishes and billing functions. The OpenFlow protocol is the most widely used control protocol in the SDN domain.

• Infrastructure layer: SDN isolates the control layer from the infrastructure layer of the network. It transfers the network intellect to a centralized controller. Consequently, the infrastructure layer now contains low end switches connected by network links. Wireless access points and the Internet are connected to the infrastructure layer. User traffic is transported through the infrastructure layer.

2. Security of SDN

SDNs are vulnerable to security threats that can be created in different units of the network. Security threats in an SDN network can be separated into four distinct types:

1) Application layer Security,
2) Control layer Security,
3) Infrastructure layer Security
4) Layer communication Security.

SDNs contain two data transfer channels: control layer data transfer and Infrastructure layer data transfer. Thus, the communication security threat can be divided into:

(1) Security relating to Control data transfer.
(2) Security relating to Infrastructure data transfer.
2.1. Security relating to Control data transfer
The core security issue in relation to Control data transfer is a lack of IP level security. Current SDN control protocols rely on higher layer secure mechanisms such as TLS (Transport Layer Security)/SSL (Secure Sockets Layer). For example, the widely used OpenFlow protocol uses TLS/SSL based control data transfer [2]. However, higher layer secure mechanisms are extremely vulnerable to IP built attacks such as IP spoofing, Denial-of-service (DoS) and reset attacks [2]. Thus, higher layer defense mechanisms are not able to deliver the required level of durability and security for the Control data transfer [2].

Furthermore, a strong authentication mechanism is required between the controller and Infrastructure data transfer. If this is absent intruders can penetrate valid Infrastructure data transfer and launch security attacks on the Control data transfer. For example, the attacker can insert fake flow requests for performance-related DoS attacks [3]. However, TLS/SSL does not perform strong authentication between layers. For example, the authentication mechanism of TLS/SSL is vulnerable to IP spoofing attacks [2].

\[
\text{Table 1. Known attacks on OpenFlow control data transfer channels.}
\]

| Attack type           | Description                                                                 |
|-----------------------|-----------------------------------------------------------------------------|
| Denial-of-service DoS | Attackers sends requests that consumes server resources to make the controller and Infrastructure unresponsive to legitimate traffic. |
| Distributed denial of service DDoS | The attacker using cipher block chaining.                                    |
| BEAST- Browser        | The attacker can recover the full plaintext when it is repeatedly encrypted in the same, or in several different, terms. |
| Exploit Against SSL/TLS attack | The attacker inserts a sequence of TCP reset requests to prematurely reset the communication session. |
| RC4 preferences in TLS | The attacker's forces change the TLS sessions allowing the data to be changed using padding at the end of the block cipher. As a result, encryption codes become less secure each time it is passed. |
| Reset attack          | The attacker uses a man-in-the-middle attack to recover the plaintext from a Cipher-block chaining encrypted TLS session. |
| POODLE attack         |                                                                           |
| LUCKY 13              |                                                                           |

The network controller is the main component of the SDN network because it comprises central intelligence and controls its functions. Consequently, attacks on the network controller represent the most serious threats to the structure of the SDN. Control data transfer is the only interface that enables a connection between the controller layer and the infrastructure layer. Thus, security in relation to data transfer control is a key factor in ensuring proper communication with the layers. A DoS attack on the SDN controller is presented in [4] in which an attacker, on an ongoing basis, sends IP packets with random headers to the controller through the control data transfer channel. This leaves the controller in an unresponsive state and unable to publish flow rules in the data transfer. TLS is used for optional menu connections in the latest open flow specification due to its complex configuration [5]. Configuration is required to generate specific network site certificates and signed machine certificates corresponding to the site-level private keys for controlling and transferring data [6]. Thus, many SDN and equipment vendors omit support for TLS in the infrastructure layer. This leaves the data channel transferring control vulnerable to security attacks. The control channel must therefore be secured using other mechanisms.

2.2. Security of the SDN data channel
The current SDN traffic architecture is unencrypted which means attackers can perform the mechanisms of an "SDN scanner" to collect network information [7]. This information can then be used to perform attacks such as DoS, reset, replay and spoofing attacks [8]. Furthermore, the current SDN data transfer channel does not contain any integrity protection mechanism. An attacker could
therefore modify the flow of change or destroy the data without being noticed by the network operator. The rotation of data flows may result in a lower quality of service (QoS) between connection layers [7]. The SDN architecture also requires powerful mechanisms for the mutual authentication of the data transmutation channel. However, hackers can impersonate these authentication mechanisms as legitimate converters and inject fraudulent traffic flows to the data level [8]. Using this method, an attacker could exhaust the flow tables and reduce the bandwidth available for the user's traffic [2]. Furthermore, it will also affect the level of control by causing unnecessary flow requests to the controller [8].

3. Elliptic Curve Cryptosystem
Elliptic curve cryptography (ECC) is one of the public key encryption algorithms and is based on the theory of an elliptical curve on limited fields. It is used to make cryptographic keys smaller, faster and more efficient. The functions and characteristics of elliptic curves have been studied in mathematics for 150 years. [9] Their use in cryptography was suggested for the first time by Neal Koblitz and then Victor Miller in 1985. [10]. ECC has received acceptance in many accredited organizations and in security protocols since the beginning of 1990 [11].

4. Elliptic Curves
Elliptic curves are mathematical constructs that have been studied by mathematicians from the seventeenth century onwards [12].

**Definition:** An elliptic curve $E$ over the finite field $F_q$ is defined by equation (1) [13]

$$y^2 = x^3 + ax + b,$$

Where $a, b \in F_q$, and $4a^3 + 27b^2 \neq 0 \pmod{q}$, together with a special point $O$, termed the point at Infinity.

The set $E(F_q)$ consists of all points $(x, y), x, y \in F_q$, which satisfy the defining equation (2), together with $O$.

**Algebraic Elliptic Curves**

The essential operation on an elliptic curve is the addition of points on the curve. To perform the addition of points on elliptic curves an algebraic formula is required. The following result provides such a formula [14].

**Theorem (Algebraic Computation Law Algorithm):**

Let $E$ be an elliptic curve given by equation (1) [15, 16]

Let $E(F_q): y^2 = x^3 + ax + b$,

Let $P_i = (x_i, y_i) \in E$.

Let $P_3 = P_1 \oplus P_2$.

Let $P_0 = (x_0, y_0) \in E$. Then: $\Theta P_0 = (x_0, -y_0)$.

If $x_1 = x_2$ and $y_1 + y_2 = 0 \pmod{q}$, then $P_1 \oplus P_2 = O$.

Otherwise, let

$$\lambda = \frac{y_2 - y_1}{x_2 - x_1}, \text{ if } x_1 \neq x_2 \quad (2)$$

$$\lambda = \frac{3x_1^2 + a}{2y_1}, \text{ if } x_1 = x_2 \quad (3)$$

$P_3 = P_1 \oplus P_2$ is given by:

$$x_3 = \lambda^2 - x_1 - x_2 \quad (4)$$

$$y_3 = \lambda(x_1 - x_3) - y_1 \quad (5)$$
5. OpenFlow protocol security methods
To increase the security of the network the security of the protocol is increased. The protocol is therefore responsible for protecting the transmission of data in a safe, robust and highly efficient way by using an Elliptic Curve Cryptography algorithm to transfer data. Efficiency and reliability will then be increased for each data transmission. The proposed method for protecting the protocol OpenFlow is to use an Elliptic curve cryptography algorithm that can encrypt and decrypt the data to be transported. Using an Elliptic curve cryptography that allows itself to encrypt and decrypt means a generation key is important as every public key and private key needs to be generated.

The sender will encrypt the message with the receiver's public key and the decryption device will then decrypt its key. This will increase efficiency and it is thus useful to have an elliptical curve algorithm that is effective in terms of file size data and encrypted files. This is achieved in the OpenFlow protocol where the protection used in TLS sessions has changed, increasing the security of the user protocol. This method will be useful for military intelligence as it facilitates transfer data by encrypting and decrypting data where only the source and destination can display information.

5.1 Encryption Algorithm
A new group will be defined for encryption. To satisfy the following condition, a and b are selected:

\[ a^2 + 4a^2 + 27b^2 \pmod{p} \neq 0 \]

In this case, a and b is smaller than p. Ep (a, b) will represent a group with pair a and b.

All pairs satisfying the above conditions will be used for encryption. In general, pairs will be found using the following method.

1. \[ 0 \leq x < p, \text{compute } x^3 + ax + b \pmod{p} \]
2. Determine whether the above result will have a square root. If it does not, x doesn’t exist in Ep. If x exists, y will have two values. Pair x and y will then be used for encryption.
   
   Then P + Q are computed as follows: \( P, Q \in E_p(a, b) \);
   
   1. \( P + O = P \)
   
   2. \( P = (x, y) \Rightarrow P + (x, -y) = O. (x, -y) \) is reverse element of \( P \).

   i.e. \( E_{23}(1,1), P = (13, 7) \Rightarrow -P = (13, 16). \)

3. \( P = (x_1, y_1), Q = (x_2, y_2), P \neq -Q \Rightarrow \text{compute} \)
   
   \[ P + Q = (x_3, y_3) \] as follows:
   
   \[ x_3 = \text{lambda}^2 - x_1 - x_2 \pmod{p} \]
   
   \[ y_3 = \text{lambda}(x_1 - x_3) - y_1 \pmod{p} \]

Therefore, lambda should be computed as follows:

\[
\text{lambda} = \begin{cases} 
  \frac{y_2 - y_1}{x_2 - x_1} & P \neq Q \\
  \frac{3x_1^2 + a}{2y_1} & P = Q 
\end{cases}
\]

5.1.1. Encryption and Decryption
- First, data G and group Ep(a, b) are produced.
- User B should produce security key nb and open Pb = nb * G as public.
- User B sends Pb to User A. User A will encrypt information Pm with Pb as follows:
- User A will produce random number K.
- \( C_m = (kG, P_m + kP_b) \)
• User A will send Cm.
• User B will decrypt Cm with security key nb as follows:
  \[ Pm + kPb - nb(kG) = Pm + k(nbG) - nb(kG) = Pm. \]

If an attacker knows G and KG, he/she can find K. This means the encryption algorithm is stronger with smaller bits.

6. **Security analysis**

This occurs when the algorithm has been applied in the emulator for networks NS3.25 and using the OpenFlow Protocol 1.3. It involves a single controller, two switches and two users in an SDN scenario. After changing the TLS sessions in original protocol and using the Elliptic curve cryptography algorithm, the second table and the third table present a comparison between the current method and the proposed method through a calculation given in units of a million instructions per second (MIPS).

| Table 2. OpenFlow Protocol using an Elliptic curve cryptography algorithm |
|---------------------------------------------------------------|
| Size of key | Computation time(MIPS) |
|------------|------------------------|
| 150        | 3.8 * 10^{10}         |
| 205        | 7.1 * 10^{18}         |
| 234        | 1.6 * 10^{28}         |

| Table 3. OpenFlow Protocol using Normal TLS |
|--------------------------------------------|
| Size of key | Computation time(MIPS) |
|------------|------------------------|
| 512        | 3.8 * 10^{4}           |
| 768        | 2 * 10^{8}             |
| 1024       | 3 * 10^{11}            |
| 1280       | 1 * 10^{14}            |
| 1536       | 3 * 10^{16}            |
| 2048       | 3 * 10^{20}            |

Thus, the algorithm is novel and strong enough to provide secure data transfer in the OpenFlow protocol used in the architecture of Software Defined Networks.

7. **CONCLUSION**

This paper addressed security challenges regarding connection data transfer channels in SDN mechanisms. It proposed a novel and secure method of data transfer from the control layer to infrastructure layer in the OpenFlow protocol, that of Elliptic Curve Cryptography. Reliability and efficiency were expected to increase for each transmission of data. The method increased network security by increasing the security of the protocol used in the Software Defined Network. This was achieved using an elliptic curve algorithm that allows itself to encrypt and decrypt the data that will be transferred and execute active classification. Any node between source and destination can be used to view information.

Finally, the security features and performance of the proposed architecture in NS3.25 simulation was analyzed. The results showed that the use of the proposed algorithm protected communication channels against attacks such as spoofing, DoS, replay, reset and eavesdropping attacks. However,
there is a penalty incurred for security performance in terms of latency and throughput. In future, the focus will be on how to leverage resources to enhance the performance of the proposed method.

References
[1] https://www.opennetworking.org/wpcontent/uploads/2015/10/Principles_and_Practices_for_Securing_Software-Defined_Networks_applied_to_OFv1.3.4_V1.0.pdf
[2] McBride, M., Cohn, M., Deshpande, S., Kaushik, M., Mathews, M., & Nathan, S. (2013). Sdn security considerations in the data center. Open Networking Foundation-ONF SOLUTION BRIEF.
[3] Kreutz, D., Ramos, F. and Verissimo, P., 2013, August. Towards secure and dependable software-defined networks. In Proceedings of the second ACM SIGCOMM workshop on Hot topics in software defined networking (pp. 55-60). ACM.
[4] Fonseca, P., Bennesby, R., Mota, E., & Passito, A. (2012). A replication component for resilient OpenFlow-based networking. Paper presented at the Network Operations and Management Symposium (NOMS), 2012 IEEE.
[5] Specification, O. S. Version 1.4. 0, October 14, 2013
[6] Zerkane, S., Espes, D., Le Parc, P. and Cuppens, F., 2016, May. Software Defined Networking Reactive Stateful Firewall. In IFIP International Information Security and Privacy Conference (pp. 119-132). Springer International Publishing.
[7] Shin, S., & Gu, G. (2013). Attacking software-defined networks: A first feasibility study. Paper presented at the Proceedings of the second ACM SIGCOMM workshop on Hot topics in software defined networking.
[8] Scott-Hayward, S., O’Callaghan, G., & Sezer, S. (2013). SDN security: A survey. Paper presented at the Future Networks and Services (SDN4FNS), 2013 IEEE SDN For.
[9] Abbas, S. A. (2015). Enhancing the Security of Identity and Access Management in Cloud Computing using Elliptic Curve Cryptography. IJERMT, 4(7), 8-15.
[10] Sudha, S., & Viswanatham, V. (2013). ADDRESSING SECURITY AND PRIVACY ISSUES IN CLOUD COMPUTING. Journal of Theoretical & Applied Information Technology, 48(2).
[11] Nida, P., Dhiman, H., & Hussain, S. (2014). A survey on identity and access management in cloud computing. Int. J. Eng. Res. Technol, 3(4).
[12] Certicom Corp., 1998 “The Elliptic Curve Cryptosystem “. A Certicom White Paper,
[13] Fibíková, L., & Vyskoc, J. (2001). Practical cryptography-the key size problem: PGP after years. Paper presented at the Proceedings in Workshop “Santa’s Get Together.
[14] Yan, S. Y. (2002). Number theory for computing: Springer Science & Business Media.
[15] Silverman, J. H. (1986). Heights and elliptic curves Arithmetic geometry (pp. 253-265): Springer.
[16] Araki, K., Satoh, T., & Miura, S. (1998). Overview of elliptic curve cryptography. Paper presented at the International Workshop on Public Key Cryptography.