A Data-origin Authentication Protocol Based on ONOS Cluster

Hua Qin and Na Wang

College of Computer Science, Beijing University of Technology, Beijing, China.

Abstract. This paper is aim to propose a data-origin authentication protocol based on ONOS cluster. ONOS is a SDN controller which can work under a distributed environment. However, the security of an ONOS cluster is seldom considered, and the communication in an ONOS cluster may suffer from lots of security threats. In this paper, we used a two-tier self-renewable hash chain for identity authentication and data-origin authentication. We analyse the security and overhead of our proposal and made a comparison with current security measure. It showed that with the help of our proposal, communication in an ONOS cluster could be protected from identity forging, replay attacks, data tampering, MITM attacks and repudiation, also the computational overhead would decrease apparently.

1 Introduction

Recently, there has been wide interest in Software-defined networking and how to secure it. A SDN controller, ONOS [1], has been designed to work under a distributed environment, and secured east-westbound protocol for the first time. There are two types of communication scenario in an ONOS cluster, where different kinds of information that need to be synchronized. One of the scenario synchronize information that needs to be strong consistent using RAFT protocol[2], and another one synchronize information that needs to be eventually consistent using gossip based anti-entropy protocol[3]. Table 1 shows the different between two communication scenarios.

Table 1. Communication scenario comparing [1]

	Communication Scenario	
	Strong Consistency	**Eventually Consistency**
Data Structure	ConsistentMap	EventuallyConsistentMap
Algorithm	RAFT	Gossip
Update Period	instantly	periodically (5ms)

In ONOS, data transfer between controllers in a cluster is default in plain text and suffer lots of vulnerability. To deal with the security challenges on control plane, ONOS has implemented TLS tunnel as a safety measure between controllers. It makes information transfer between two controllers confidentially. But it still has some problems, and it can be described as follows:

First of all, current TLS transmission between ONOS controllers is fully encrypted. One cannot implements different security policy for different communication scenarios.

In the strong consistency communication scenario, controllers keep information consistent with the leader controller. Attacker can hijack information sent by leader and make attacks which may result in fraudulent data be preserved in every controller in the cluster and even may cause the data plane impossible to use. Thus in this scenario encryption is a must. By contrast, in the eventually consistency communication scenario, where each ONOS controller has an equal role, and sends information more frequently than strong consistency communication (every 5ms). In this scenario, encryption is not necessary because of the characteristic of gossip protocol. Even if an attacker tampers information and sends to an ONOS controller, correct information can be received from other controller. If TLS is implemented, information will become consistent once it is received, but the encryption and decryption will become the performance bottleneck of the controller when a cluster has a large scale. On the contrary, transfer information in plain text in this scenario will greatly decrease the system cost, while the time when information become consistent will be longer. Due to the characteristic of gossip protocol, information transferred in this scenario is not time sensitive, lower cost security methods like integrity verification and digital signature are enough to protect information from attacks. Therefore in an ONOS cluster, different communication scenarios need different security policies.

Secondly, communication between ONOS controllers doesn't have the mechanism of data-origin authentication. An ONOS controller keeps listening a specific port, and with no authentication an attacker can impersonate an controller in a cluster and send fraudulent data to the cluster easily, which would cause severe problem on the data forwarding plane. For this reason, a light-weight authentication is needed for east-westbound communication, which secure ONOS controllers from malicious attacks.

What's more, TLS [4] tunnel itself is prone to security threats. In our scenario, MITM attacks would take place and both parties of communication could deny the message they have sent [5]. During the handshaking phase of TLS session, all message is sent in plain text, so an attacker can get these information easily, and use them to build a TLS tunnel with a valid controller. Moreover, the secret session key for TLS tunnel is shared by both parties, TLS tunnel cannot assure non-repudiation.

In order to prevent these problem, we proposed a light-weight data origin authentication protocol based on a two-tier hash chain on ONOS controllers. The main purpose of our proposal is to secure information in an ONOS cluster from eavesdropping, tampering and malicious controller. This protocol can be a supplemental security measure for the current TLS tunnel between ONOS controllers, also can provide sufficient security for eventually consistency communication scenario when message transfers in plain text.

We have organized the rest of this paper in the following way. In section two, we introduce the background and related work of our proposal. Section three describes the protocol in detail. And in section four, we analyze the security and overhead of our proposal. Finally in section five, we present our conclusion.

2 Background and Related Work

Digital signature can be used for data-origin authentication. The key technology of digital signature are: public key cryptography, shared key cryptography and Hash algorithms. Compare with the first two technology, hash algorithms demonstrated high reliability and high efficiency.

Due to the frequent traffics between ONOS controllers in a cluster and high computation overhead caused by public key and shared key cryptography, an authentication method with low computation cost and high efficiency is needed. It follows that Hash algorithm is more suitable for communication in an ONOS cluster and capable to meet the security requirements. Hash chain is a method widely used for data-origin authentication [6], and is capable of securing information in an ONOS cluster from tampering, data replay and repudiation.

To work around the length limitation of tradition hash chain [7], the Self-Renewal Hash Chain [6] has been proposed. This new kind of hash chain can be self-renewing without addition protocols and re-initialization smoothly. Another approach, the Two-tier One-way Hash Chain [8], have gave significant performance improvement over previously proposal. This proposal has a single first-tier chain and multiple second-tier chains. The first-tier hash chain offers seeds for second-tier chains which is computed by the first-tier hash function. And second-tier chains are used for authenticating the transaction between the server and clients. These methods based on hierarchical hash chains are more suitable for the communication scenarios.

Recent research on SDN has been focusing on securing east-westbound communication [9, 10]. Jun-Huy Lam et al proposed a scheme of securing the distributed SDN communication with a multi-domain capable Identity-Based Cryptography (IBC). This scheme uses symmetric session key to encrypt communication between two controllers. Study in [10] also used IBC to authenticating the identities of controllers and shared a session key for east-westbound communication. Both of studies in studies in [9, 10] provide a way to communicate encrypted in a SDN cluster. But they still cannot protect SDN controller from illegal controller which will eavesdrop, tamper or replay the traffic in a cluster. Moreover, these schemes depends on symmetric encryption which will cause massive computation cost.

3 A Data-origin Authentication Protocol Based on ONOS Cluster

In this section, we propose a two-way data-origin authentication protocol for ONOS cluster. Our proposal is based on two-tier self-renewable hash chain introduced by Alabrah et al [8] and Zhang et al [6] which are not meant for SDN. In our proposal, two tiers of hash chain, as shown in Fig.1, are used for different purposes. The first tier of hash chain is used for identity authentication between two ONOS controllers and also for providing seed for second tiers at the beginning of the communication in an ONOS cluster. And second tiers of hash chain are used for data-origin authentication during the communication.

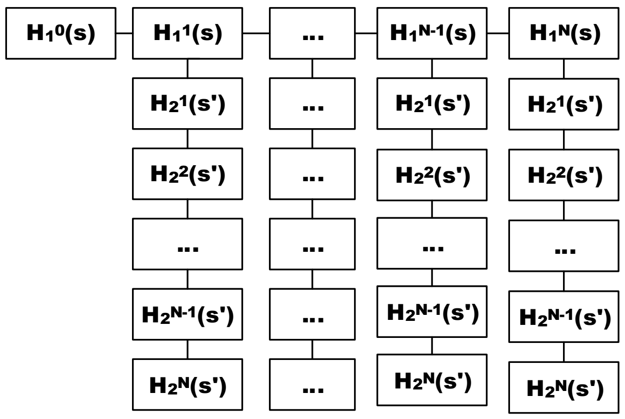

Figure 1. Two-tier self-renewal hash chain

The whole protocol is done in three routines: identity authentication, data-origin authentication and seed update. For eventually consistency communication scenario, all three steps are needed. For strong consistency communication scenario, only data-origin authentication and seed update are needed because TLS already offers identity authentication. Using the notation presented in Table 2, we describe the main components of the proposal in order to communicate.

Table 2. Notation

h_x	Hash function of the x tier hash chain
S_X	Seed of ONOS controller X
PKX, SKX	Private and public key of ONOS controller X
MAC	Integrity check code
N_{XY}	Length of the y tier of hash chain of ONOS controller X
E	Encryption
rX	Random number produced by ONOS controller X
s	Serial number of packets
T_X	Data-origin authentication token from ONOS controller X

3.1 Identity Authentication

The identity authentication routine contains three steps: *Initialization*, *Message transaction* and *Verification*, and two participants which are denoted by controller A and controller B, as shown in Fig. 2.

Initialization is used to distribute the information for identity authentication and seed of the first tier hash chain. Before cluster deployment, the deployment server which works as a Trusted Third Party (TTP) distributes the private and public key for identity authentication and the seed of first tier hash chain. Also the deployment server of the ONOS cluster will send a cluster description file to every ONOS controller in the cluster which contains all ONOS controllers' ID and their states. After ONOS controllers receive everything mentioned above, each controller computes the first hash chain and broadcasts the tail of first hash chain to every controllers in the cluster. In what follows, we give the notations of the identity authentication routine.

ID_A = Identity code for controller A
ID_B = Identity code for controller B
N_1 = length of first tier hash chain
N_2 = length of second tier hash chain
IDSet = controllers' ID set in the cluster description file
mac = hash function
M = message payload between hash chain node and MAC
H = authentication token

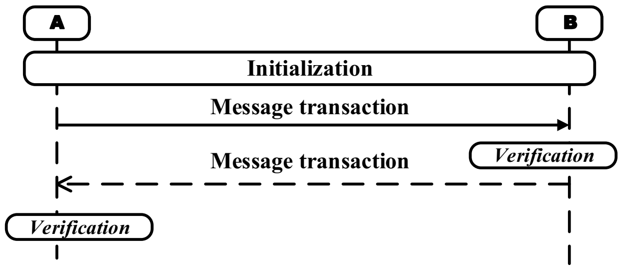

Figure 2. Identity authentication steps

Then, every ONOS controller begins *Message transaction* and *Verification*. Fig. 3 illustrates the messages transform between controllers in detail. During *Message transaction*, controllers select and exchange a random number to authenticate the identity of each other and exchange the tail of second hash chain for later data-origin authentication after they pass the identity authentication.

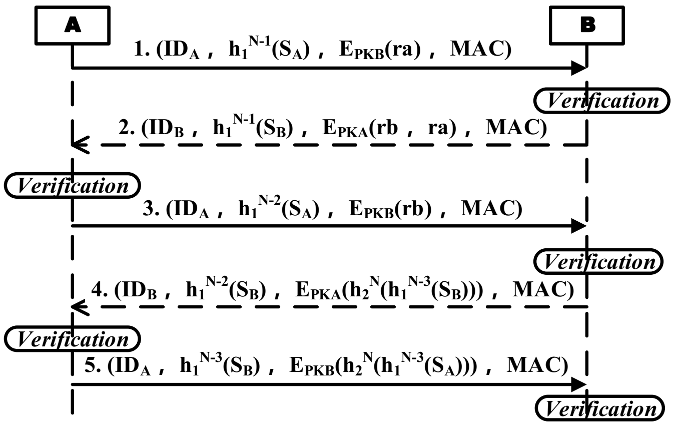
A
B
1. (ID_A , $h_1^{N-1}(S_A)$, $E_{PKB}(ra)$, MAC)
Verification
2. (ID_B , $h_1^{N-1}(S_B)$, $E_{PKA}(rb , ra)$, MAC)
Verification
3. (ID_A , $h_1^{N-2}(S_A)$, $E_{PKB}(rb)$, MAC)
Verification
4. (ID_B , $h_1^{N-2}(S_B)$, $E_{PKA}(h_2^N(h_1^{N-3}(S_B)))$, MAC)
Verification
5. (ID_A , $h_1^{N-3}(S_B)$, $E_{PKB}(h_2^N(h_1^{N-3}(S_A)))$, MAC)
Verification

Figure 3. Message transaction step

After receiving message from other controllers, every controller needs to do *Verification* to check the massage is valid or not. Follows are the *Verification* routine, in which payload is all encrypted information during *Message transaction*.
For the kth verification on controller B:

```
Verification ($ID_A$,H,Payload,MAC)
Begin
If ($ID_A$ is in IDSet) then
    $T := TA$
    If ($T == h_1(H)$) then
        $MAC' := mac(ID_A, H, Payload)$
        If ($MAC' == MAC$) then
            $T_A := H$
            $N_{A1} := N_{A1}-1$; end_if
    end_if
end_if
if($N_{A1}$==0) then
    $T_A$:= Call Seed_Update()
    $N_{A1}$:=$N1$; end_if
Return (TRUE);
End
```

3.2 Data-origin Authentication

When controllers finish identity authentication, consistency communication begins. And each controller need to sign on the message it sends and let others do data-origin authentication to check message sender is valid or not. Data-origin authentication routine is shown in Fig. 4, where S'_A and S'_B is the next token of the first hash chain from the sender. Each time a controller receives a message, it checks serial number and token of the second tier hash chain by *Verification*. The *Verification* routine is as same as we mentioned above, and the Payload is the serial number of the message and payload it carried. What's different is that at beginning of data-origin routine T_A and T_B equals to the tail of each controller's second hash chain.

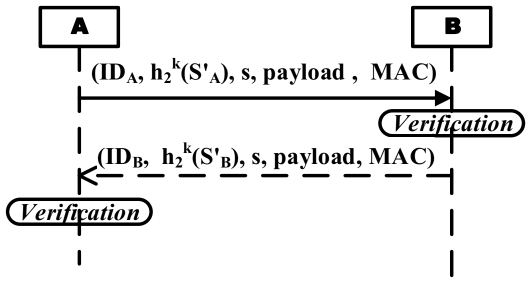
A
B
(ID_A, h_2^k(S'_A), s, payload , MAC)
Verification
(ID_B, h_2^k(S'_B), s, payload, MAC)
Verification

Figure 4. Data-origin authentication routine

3.3 Seed Update

When the hash chain is going to use up, the seed update routine is needed. For the first tier hash chain, the seed is updated by a right shift operation. And for the second tier hash chain, the new seed equals to the first unused token of the first hash chain. For example, controller B is going to do the kth time of seed update, the new seed is h_1^k+5(S_B). After update the seed, new tail of hash chain is carried by the Message transaction to the controllers in the cluster for later authentication. Fig. 5 shows the message that carries new tail of hash chain, where S'new and Snew are respectively the new seeds of second tier hash chain and first tier hash chain.

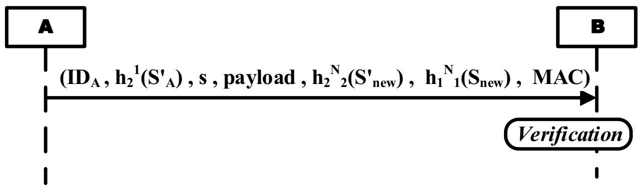
A
B
(ID_A , h_2^1(S'_A) , s , payload , h_2^N_2(S'_new) , h_1^N_1(S_new) , MAC)
Verification

Figure 5. Message carries new tail of hash chain

4 Analysis

4.1. Security analysis

In this section, we analysis the security of the data-origin authentication protocol we proposed. The proposed scheme is a light-weight protocol and can provide several security assurances which are analysis in detail.

Unforgeability of controller's identity: In this proposal, the seed which is used to calculate the hash chain is securely sent from the TTP. Thus only a valid controller owns the seed and the hash chain generated by the seed. And because of the irreversible of the hash function, attacker who captures message between two valid controllers and get the token, cannot deduce the seed from a token. Therefore the identity of a valid controller cannot be forged.

Protection against MITM attacks: Suppose an attacker uses MIMT attack to get the certificates of valid controllers and creates a TLS tunnel with them. Because of lacking of the token, message sent by the attacker cannot pass the data-origin authentication, and a valid controller will notice that there is an attack, then cuts the connection.

Availability of data: The proposal provides serialization and integrity verification for every message transferred between controllers. Consequently, replay attacks and data tampering are avoided.

Non-repudiation: In this proposal, only valid controllers can provide tokens, thus a controller cannot deny the message it sent.

4.2. Overhead Analysis

We analysis performance overhead of our protocol in detail, and compare with tradition TLS protocol. We present a detailed analysis of the overhead associated with the identity authentication routine and data-origin authentication. T_{ASYM} and T_{SYM} are computational overhead for encryption or decryption using asymmetric algorithm and symmetric algorithm respectively. Thash is computational overhead for hash function. And T_{CERT} is computational overhead for certificate verification.

During the identity authentication routine, each controller costs $5T_{ASYM}$ +$3T_{hash}$ computational overhead. And during data transfer, for each transaction, a controller only costs 2Thash computational overhead because of data-origin authentication. Compare with traditional TLS protocol, which will cost T_{CERT}+$2T_{ASYM}$ during TLS handshake for two-way identity authentication, and T_{SYM} + T_{hash} computational overhead during data transfer. Table 3 shows the overhead comparison between our protocol and traditional TLS protocol.

Table 3. Comparison between our protocol and traditional TLS protocol

Protocol	Overhead during identity authentication	Overhead during data transfer
Our Protocol	$5T_{ASYM}$ +$3T_{hash}$	$2T_{hash}$
TLS Protocol	T_{CERT}+$2T_{ASYM}$	T_{SYM} + T_{hash}

According to the analysis in studies [11, 12], the computational overhead of encryption or decryption using asymmetric algorithm or symmetric algorithm, which are T_{ASYM} and T_{SYM}, is much higher than the computational overhead of hash function, which is T_{hash}. In addition, Thash of the commonly used hashing algorithms, such as MD5 and SHA, is usually quiet low. Thus, when we use the proposed protocol as a supplemental security measure for the current TLS tunnel, the computational overhead would rise slightly. Also, when we use is as the security measure for eventually consistency communication scenario, the computational overhead would decrease apparently than current TLS tunnel.

5 Conclusion

We proposed a two-way light-weight data-origin authentication protocol for ONOS cluster, which is based on two-tier self-renewable hash chain. The proposal we designed can be used as a supplemental security measure for the current TLS tunnel between controllers in an ONOS cluster, also can provide sufficient security for eventually consistency communication scenario. The protocol can protect controllers from attacks such as MIMT, data tampering, replay, and identity forge. Also the protocol can provide non-repudiation for controllers, which controllers cannot deny the messages they send. As future work, we envision an improvement of the protocol from inter-domain to multi-domain.

References

[1] Information on: https:/wiki.onosproject.org
[2] Information on: http://thesecretlivesofdata.com/raft/
[3] Information on: https://wiki.onosproject.org/display/ONOS/Network+Topology+State#NetworkTopologyState-FailureHandlingandAnti-Entropy
[4] Dierks T, Rescorla E. The Transport Layer Security (TLS) Protocol Version 1.1. In: E R, editor. (2006)
[5] Information on: http://www.ntu.edu.sg/home/ehchua/programming/webprogramming/HTTP_SSL.html
[6] Haojun Z, Xiaoxue L, Rui R. A Novel Self-Renewal Hash Chain and Its Implementation. 2008 IEEE/IFIP International Conference on Embedded and Ubiquitous Computing. Shanghai, p.144-149(2008)
[7] Xiao-Yuan Y, Jia-Yong C, Jing-Jing W. A Self-Renewal Hash Chain Scheme Based on Fair Exchange Idea(SRHC-FEI). 3rd IEEE International Conference on Computer Science and Information Technology. Sichuan, p.152-156(2010)
[8] Alabrah A, Bassiouni M. A Hierarchical Two-Tier One-way Hash Chain Protocol for Secure Internet Transactions. Anaheim, CA, p.868-873(2012)
[9] Jun-Huy L, Sang-Gon L, Hoon-Jae L, Eko OY. Securing Distributed SDN with IBC. ICUFN 2015. Sapporo, p.921-925(2015)
[10] Santos MAS, Nunes BAA, Obraczka K, Turletti T, de Oliveira And Cintia B. Margi BT. Decentralizing SDN's Control Plane. 39th Annual IEEE Conference on Local Computer Networks. Edmonton, AB, p.402-405(2014)
[11] PAULSON LC. Inductive Analysis of the Internet Protocol TLS. ACM Transactions on Information and System Security. **2,** 3, p.332-351(1999)
[12] Apostolopoulos G, Peris V, Saha D. Transport Layer Security: How much does it really cost. INFOCOM '99. Eighteenth Annual Joint Conference of the IEEE Computer and Communications Societies. New York, p.717-725(1999)