Authenticated Group Key Management Scheme in Service Chain

Jiang Hua\textsuperscript{a}, Wang Qingrui\textsuperscript{a,b}, Fan Jinping\textsuperscript{c}, and Zhang Gang\textsuperscript{d}

Beijing Electronics Science and Technology Institute, No. 7, Fu Feng Road, Beijing, China
\textsuperscript{a}jh@besti.edu.cn, \textsuperscript{b}wqr_sdlc@163.com, \textsuperscript{c}fanjinpo@yeah.net, \textsuperscript{d}Super_zgang@126.com

\textbf{Keywords:} service chaining, group key, authentication, virtual network functions

\textbf{Abstract:} The service chain, which is not dependent on the special hardware facilities and the network topology is changeable, is studied, and an authenticated group key management scheme suitable for service chain is proposed. The scheme is based on the bilinear mapping cryptosystem and combines the threshold idea with the identity authentication method, which improves the efficiency and security of the protocol. This scheme also realizes the connection security between the virtual network functions in the service chain while carrying out group key updating, and proves its correctness and security. The analysis results show that the scheme is suitable for the dynamic key management of service chain with the advantages of small number of wheels and small computing overhead in ensuring the safety of each instance in the service chain.

1. Introduction

Currently, operators need to use dedicated hardware devices when deploying network functions such as deep packet inspection, caching, traffic optimization, and firewalls. These hardware devices may come from different operators, making management more difficult. At the same time, the definition of the service chain is static in the existing network infrastructure, which depends on the topology of the network. Operators need to redeploy hardware devices when adding or removing network functions, resulting in increased operating costs and expenditure costs. Network Function Virtualization (NFV) redefines the typical network function delivery and operation methods. By using standard IT virtualization and cloud technologies, NFV defines a structure in which network functions and applications are implemented only as software entities and are independent of hardware. [1] The scheme proposed a high-level framework for NFV. By decoupling the software from dedicated hardware, NFV can abstract these traditional network functions into virtual network functions (VNFs). And by managing and controlling VNFs in the management and orchestration layer, NFV can generate the required network service which is service chain. Based on NFV, it can overcome the shortcomings of the traditional static service chain and realize the dynamic, on-demand, and fast construction of the service chain. [2] This scheme introduces a customizable function chain. It simulates the service and variability of the service chain and proposes a management framework for deploying these services, which takes into account the runtime variability of the service chain. [3] It discusses the dynamic implementation of the VNF chain on the OpenStack cloud platform in the standard industry, which focuses on the complexity of the
underlying virtual network infrastructure. [4] The scheme proposes a hierarchical service function chain framework. It divides the service function chain into multiple layers by using software defined network (SDN) and implements this architecture through OpenDaylight. The service chain is an important feature of NFV, which creates flexible environment in NFV, but it also brings new security issues. Compared with traditional network services, virtualization technology has been introduced into the service chain, so the service chain also faces virtualization security issues. Each service node in the service chain is a VNF instance. When a VNF instance is dynamically added or deleted in the service chain, that is, when the topology changes, it is necessary to consider the problem of trust reconstruction between the nodes in the service chain and the dynamic, efficient and real-time nature of the service chain. At the same time, because data packets are forwarded between adjacent VNF instances, transmission security issues are faced between VNF instances.

In view of these characteristics of the service chain and the security issues it faces above, this paper proposes a threshold group key management scheme based on the identity cryptosystem. It uses the idea of bilinear mapping[5] and secret sharing to ensure forward security and backward security of the service chain, and performs identity authentication between adjacent VNFs[6].

2. Related Basic Knowledge

2.1. Bilinear Map

Let $G_1$ be a cyclic additive group, whose order is a prime $p$, and $G_2$ be a cyclic multiplicative group with the same order $p$. The bilinear map is given as $e: G_1 \times G_1 \rightarrow G_2$.

Which satisfies the following properties:

(1) Bilinearity: $e(aP_1, bP_2) = e(P_1, P_2)^{ab}$, for all $P_1, P_2 \in G_1$, $a, b \in \mathbb{Z}_p^*$. 

(2) Non-degeneracy: There exist $P_1, P_2 \in G_1$ such that $e(P_1, P_2) \neq 1$, in other words, the map does not send all pairs in $G_1 \times G_1$ to the identity in $G_2$. 

(3) Computability. There is an efficient algorithm to compute $e(P_1, P_2)$, for all $P_1, P_2 \in G_1$.

The security foundation of this scheme is the computable Diffé-Hellman (CDH) problem. That is, arbitrary $a, b, c \in \mathbb{Z}_q^*$, $P_1, P_2, P_3 \in G_1$, it is difficult to determine $e(P_1, P_2, P_3)^{abc}$.

2.2. (t, n) Threshold Scheme

Threshold is an important theory in cryptography. It is a secret sharing technology, which the key is shared by multiple members. Let $p$ be a prime number. The system assigns $n$ members a share of secret $d$ (part secret) process.

(1) The $t-1$ polynomial $h(x) = d + a_1x + a_2x^2 + \cdots + a_{t-1}x^{t-1}$ on $F_q$ is selected randomly to satisfy $h(0) = d$ .

(2) Selecting $n$ non-zero, two-different elements in $F_q$, where $d_i = h(c_i)$. Let $(c_i, d_i)$ be the share of $d$ that is kept secretly by the i-th member, $i = 1, 2 \cdots n$.

Any system member greater than $t$ may provide at least $t$ shares of secret $d$: $(c_{i_1}, d_{i_1}), (c_{i_2}, d_{i_2}), \cdots (c_{i_t}, d_{i_t})$. At this time, the polynomial can be reconstructed by Lagrange interpolation, shown in Equation (1).
\[ h(x) = \sum_{s=1}^{t} d_s \prod_{i<j \in s} \frac{x-c_i}{c_i-c_j}. \]  

(1)

\[ d = h(0) = \sum_{s=1}^{t} d_s \prod_{i<j \in s} \frac{-c_i}{c_i-c_j}. \]  

(2)

But any less than \( t \) system members cannot obtain any information about the secret \( d \).

3. Identity-based Service Chain Key Management Scheme

3.1. System Initialization

This scheme uses distributed key management [7]. During the initialization process, the VNFM can distribute IDs for each VNF instance and VNFM is an interim management center. Suppose there are \( n \) service nodes in the service chain, each service node corresponds to a unique identifier \( \text{id}(i_{ID}, \ldots, i_{ID}) \), \( i = 1, 2, \ldots, n \). Simultaneously it defines 4 hash functions: 

\[ H_1 : \{0,1\}^l \rightarrow G_1, \quad H_2 : G_1 \rightarrow F_q^*, \quad H_3 : G_2 \rightarrow \{0,1\}^l \] 

(The length of the communication key is \( l \) in the service chain), 

\[ H_4 : \{0,1\}^* \rightarrow F_q^*. \]

Initialization process:

Step 1: VNFM generates system parameters on the service chain, such as 
\[ e : G \times G = G, \quad q, \quad G, \quad G_2 \]. VNFM randomly selects \( d \in F_q^* \) as the private key of the service chain, \( Q = dP \) as the public key. VNFM passes system parameters to service nodes through multicast.

Step 2: 
\[ f(x) = d + a_1 x + a_2 x^2 + \cdots + a_n x^n \] 

is \( (P, Q, H_1, H_2, H_3, H_4) \) randomly generated by VNFM, where \( f(0) = d \). The VNFM generates a system private key \( D_i \) for each service node \( N_i \) and transmits \( D_i \) to \( N_i \) through the secure channel.

Step 3: The private key \( D_i = dH_3(ID_i) \) of the service node \( N_i \) is calculated by VNFM and \( D_i \) is transmitted to the service node through the secure channel.

Step 4: The role of the VNFM as temporary management center disappears and it no longer participates in other operations in the service chain.

3.2. Communication Key Generation Protocol in Service Chain

Step 1: The service node \( N_i \) randomly selects \( r \in F_q^* \) as the private parameter of the service node. Then it calculates the equation: 
\[ W_i = H_1(ID_i), \quad T_i = rP, \quad P_i = H_2(e(W_i, T_i))D_i \]. After the calculation is completed, it will send \( (T_i, P_i) \) to its adjacent service node \( N_{i-1} \) and \( N_{i+1} \) (where if \( i = 1 \), \( i-1 = n \); if \( i = n \), \( i+1 = 1 \)).

Step 2: When the service node \( N_i \) receives the message \( (T_j, P_j) \) of the service node \( N_{i-1} \) and \( N_{i+1} \), it verifies that the equation 
\[ e(W_i, P_j) = e(D_i, H_j(e(W_j, T_j)W_j)) \] 
is or not correct. If it corrects, then it
calculates the equation \( E_i = e(T_{i+1} - T_{i-1}, Q)^Y \) \( V_i = H_2(E_i)D_i \) (where \( i = 1 \), \( i - 1 = n \); if \( i = n \), \( i + 1 = 1 \)). After the calculation is completed, the service node \( N_i \) broadcasts \((V_i, E_i)\) to other service nodes.

Step 3: After the service node \( N_i \) receives the broadcast message \((V_j, E_j)(j = 1, 2, \cdots, n; j \neq i)\) from the service node \( N_j \), it verifies that the equation \( e(W_iV_j) = e(D_jH_2(E_j)W_j) \) is or not correct. Then each service node can calculate the group communication key of the service chain system according to the equation:

\[
K = H_3(e(nrT_{i-1}, Q)E_i^{e^{-1}}E_{i-2}^{-1} \cdots E_{n-1}E_1^{-1}E_{n-2}^{-1} \cdots E_{i+1}^{-1})
\]

### 3.3. Joining a New Service Node

The newly added service node is \( N_{n+1} \), and the Identity Identifier is \( ID_{n+1} \). Update the service chain communication key.

**Step 1:** The service node \( N_{n+1} \) randomly selects \( r_i \in F_q^* \) as the private parameter of the service node and it calculates the equation \( W_{n+1} = H_1(ID_{n+1})T_{n+1} = r_{n+1}P \), \( r_{n+1} \in F_q^* \). Then \( t \) service nodes are selected randomly in the service chain, and the \( t \) service nodes calculate the equation \( Y_i = e(r_{n+1}Q, W_i) \). After the calculation is completed, service node \( N_{n+1} \) will send \((Y_i, T_{n+1})\) to service node \( N_i \).

**Step 2:** After \( t \) service nodes receive the registration information of service node \( N_{n+1} \), they perform identity verification on \( N_{n+1} \), that is, it verifies the equation \( Y_i = e(T_{n+1}, D_i) \) is correct or not. If the identity verification fails, the new service node is refused to join. If passed, then the service chain is allowed to join, and the communication key of the service chain needs to be updated, and the master key component needs to be safely generated for the new service node.

**Step 3:** The private key component of the service node \( N_{n+1} \) is calculated by the \( t \) service nodes which are participating in the identity audit, shown in Equation (3).

\[
D_{n+1,i} = d_iH_i(ID_{n+1}) \prod_{j=1}^{t} \frac{ID_j}{ID_j - ID_i}
\]

System key components stored by service node \( N_{n+1} \) is shown in Equation (4).

\[
d_{n+1,i} = d_i \prod_{j=1}^{t} \frac{ID_{n+1} - ID_j}{ID_j - ID_i}
\]

**Step 4:** \( t \) service nodes calculate the equation \( kh = H_3(e(Q, H_4(K^{'i})) \), \( k_i = H_3(e(W_{n+1} + W_j, T_{n+1})^Y) \), \( C_i = \{D_{n+1,j} \parallel d_{n+1,i} \parallel kh\}_{k_i} \), \( R_i = H_4(C_i, H_i(k_i)) \), where \( K^{'i} \) is the communication key of the current service chain. After the calculation is completed, service node \( N_i \) will send \((R_i, C_i, T_i)\) to service node \( N_{n+1} \). Where \( \parallel \) represents a connector, \( \{\cdots\}_{k_i} \) means that the information is encrypted with \( k \).

**Step 5:** When the service node \( N_{n+1} \) receives the message \((R_i, C_i, T_i)\) of the service node \( N_i \), it calculates the equation \( k_i = H_3(e(W_{n+1} + W_j, T_i)^Y) \) and it verifies the equation \( R_i = H_4(C_i, H_i(k_i)) \) that is
correct or not. If it corrects, it can get $D_{a+1}, d_{a+1}, kh$ by decrypting the cipher text $C_i$ with $k_i$. If the information sent by some service nodes is not received correctly, the request is sent to other service nodes again until the service node $N_{a+1}$ successfully receives the correct message from the service node. And it calculates the following parameters:

1. The private key of service node $N_{a+1}$: $D_{a+1} = \sum_{i=1}^t D_{a+1,i}$.
2. The system key component $kh = H_i(e(Q,H_i(K'))) \mod t$ of the service node $N_{a+1}$: $d_{a+1} = \sum_{i=1}^t d_{a+1,i}$.
3. The communication key of the new service chain: $K_{new} = kh \otimes kc$, where $kc = H_i(e(W_{a+1},r_{a+1}P))$.
4. $M = \{kc\}_{kh}, U = r_{a+1}W_{a+1}, V = (r_{a+1} + H_i(M,W_{a+1}))D_{a+1}$.

Step 6: The service node $N_{a+1}$ broadcasts $(U,V,M)$ to all service nodes in the service chain. When the service node $N_i$ receives the message $(U,V,M)$ of the service node $N_{a+1}$, it verifies the equation $e(P,V) = e(Q,U + H_i(M,W_{a+1}))W_{a+1}$ is correct or not. If it corrects, it calculates the equation. And it can get $kc$ by decrypting $M$ with $kh$, finally it can calculate the communication key for the new service chain $K_{new} = kh \otimes kc$.

### 3.4. Service Node Departure or Periodic Key Update

When service nodes remain unchanged for a long time or a service node leaves the service chain, the key of service chain needs to be updated. At this point, it is initiated by any service node (assuming service node $N_i$) in the service chain. Follow these steps:

Step 1: The service node $N_i$ randomly selects $r_i \in F_{q_i}^*$ as the private parameter of the service node. Then it calculates the equation $kh = H_i(e(Q,H_i(K'))) \mod t$, where $K'$ is the communication key of the current service chain.

Step 2: All service nodes in the current service chain except $N_i$ sends $(T_j,P)(j \neq i)$ to service node $N_i$. Then service node $N_i$ verifies the equation $e(W_i,P) = e(D_i,H_i(e(W_j,T_j)W_j))$ that is correct or not.

Step 3: The service node $N_i$ calculates the equation $kc = H_i(e(r,P,W_i)), U = rW_i, V = (r_i + H_i(M,W_i))D_i$. After the calculation is completed, service node $N_i$ will send $(U,V,M)$ to other service node.

Step 4: When the service node $N_j$ receives the message $(U,V,M)$ of the service node $N_i$, it verifies the equation $e(P,V) = e(Q,U + H_i(M,W_j))W_j$ that is correct or not. If it corrects, it calculates the equation $kh = H_i(e(Q,H_i(K')))$.

Step 5: The new group communication key is: $K_{new} = kh \otimes kc$. 

159
4. Analysis of the Scheme

4.1. Correctness Analysis

This scheme proves its correctness by updating the key from three aspects: key generation, new service node joining, and service node leaving.

Proposition 1. In the process of generating the group key, each service node can correctly calculate the group key.

Proof: According to the nature of the bilinear map $e(aP, bP') = e(P, P')^ab$, it can calculate Equation (5) and (6).

\[ E_i = e(T_{i+1} - T_{i+2}, Q) \]
\[ = e((r_{i+1} - r_{i+2})P, sP)^{r_{i+1} - r_{i+2}} \]
\[ = (P, P)^{(r_{i+1} - r_{i+2})r^{r_{i+1} - r_{i+2}}} \]
\[ K = e(n_rT_{i+1}, Q)E^{n-2}_{i+1}E^{n-2}_{i+1} \cdots E^{n-2}_{i+1}E^{n-2}_{i+1} \cdots E^{n-2}_{i+1} \]
\[ = e(P, P)^{n_r,E^{n-2}_{i+1}}E^{n-2}_{i+1} \cdots E^{n-2}_{i+1}E^{n-2}_{i+1} \cdots E^{n-2}_{i+1} \]
\[ = e(P, P)^{n_r,E^{n-2}_{i+1}(t_{i+1} - t_{i+2})} \cdots E^{n-2}_{i+1}(t_{i+1} - t_{i+2}) \]
\[ = e(P, P)^{n_r,E^{n-2}_{i+1}(t_{i+1} - t_{i+2})+E^{n-2}_{i+1}(t_{i+1} - t_{i+2})+ \cdots} \]

Therefore, each service node calculates the same group key $K$.

Proposition 2. When a new service node joins a service chain, it performs a group key update on the service chain, and the final calculated group key is $K_{new}$.

Proof: For the newly added service node $N_{new}$, it can calculate the equation $kc = H_2(e(W_{new}, r_{new}, P))$. After authentication, based on the message $T_i$ received from $N_i$, it can calculate the equation $k_i = H_j(e(W_{new} + W_i, T_i)^{e_{new}})$. The service node $N_{new}$ decrypts $C_i = \{D_{i+1} \| d_{i+1} \| \| \}$ with $k_i$ to get $kh$, so the communication key of the new service chain for service node $N_{new}$ is available. For other service nodes on the service chain, they can calculate the equation $kh = H_j(e(Q, H_j(K')))$. Then it calculates the equation $K_{new} = kh \oplus kc$. Therefore, each service node calculates the same group key $K_{new}$.

Proposition 3. When the service node leaves the service chain, the group key is updated, and new group key is $K_{new}$.

Proof: After a service node leaves the service chain, the service node $N_i$ that initiated the key update in the service chain calculates the equation $kc = H_3(e(r_iP, W_i))$. According to the equation $kh = H_j(e(Q, H_j(K')))$, the new service chain communication key is available for service node $N_i$. After the identity of other service nodes $N_j$ in the service chain is determined by $N_i$, the service node $N_j$ can decrypt $M = \{ke\}_{e_{new}}$ by calculating the equation $kh = H_j(e(Q, H_j(K')))$. Finally, the communication key calculated by all service nodes on the service chain is the same.

4.2. Security Analysis

The security of this protocol is based on Diffie Hellman's difficult problem. The master key in
the protocol must be generated by the user. Each service node only has its own secret share, and cannot obtain the secret shares of other service nodes, thus improving master key security in the service chain. And the communication key in this protocol is independent because the generation of communication key is related to random parameters $t^i \in F^*_q$.

1) Forward security
The newly added service node can obtain $kh$ by decrypting the ciphertext $C_i$, and $kh = H_i(e(Q, H_i(K^i)))$ is obtained from the hash function. The hash function is unidirectional and the new node cannot obtain the group communication key before it joins in the service chain, which ensures the forward security of the service chain.

2) Backward security
After the service node leaves, it can still calculate $kh$, but it can’t satisfy the equation $e(W_i, P_j) = e(D_j, H_j(e(W_j, T_j)W_j))$. And it can’t receive the message from the service node $N_i$. Therefore, it can’t obtain the updated group communication key, ensuring the backward security of the solution.

4.3. Performance Analysis

In terms of communication overhead, the protocol requires 4 single-point communications and 2 broadcast communications during the initialization and key generation phase. It requires 2 single-point communications and 1 broadcast communications during the new service node join phase. During the departure phase of the service node, it requires (n-1) single-point communications and 1 broadcast communications. As shown in Table 1, it is described that the protocol is compared with other key protocols[8-10] with members joining and exiting (where n represents the number of group members and h is the height of the tree). From Table 1, it can be seen that although this scheme has higher computational complexity than the $TGDH$ protocol, it has the advantage of fewer rounds and fewer broadcasts.

<table>
<thead>
<tr>
<th>Protocol name</th>
<th>communication</th>
<th>Computational complexity</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Number of rounds</td>
<td>Number of broadcasts</td>
</tr>
<tr>
<td>Certification Protocol $BD$</td>
<td>Node join</td>
<td>2(n+1)</td>
</tr>
<tr>
<td></td>
<td>Node exit</td>
<td>2(n-1)</td>
</tr>
<tr>
<td>$TGDH$ Protocol</td>
<td>Key generation</td>
<td>$h$</td>
</tr>
<tr>
<td></td>
<td>Node join</td>
<td>2</td>
</tr>
<tr>
<td></td>
<td>Node exit</td>
<td>1</td>
</tr>
<tr>
<td>The scheme</td>
<td>Key generation</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>Node join</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>Node exit</td>
<td>3</td>
</tr>
</tbody>
</table>

5. Conclusion

Based on the characteristics of service chain, this paper proposes a group key management scheme suitable for dynamic service chain. The scheme realizes the security of the service node by performing identity authentication between adjacent VNFs in the service chain. Compared with other group key management schemes, this scheme can achieve rapid update of group keys with low overhead. Although the new scheme increases the number of unicasts, the number of communication rounds is smaller as well as the number of broadcasts. In terms of computational
complexity, the scheme uses a symmetric encryption algorithm, and the complexity of the calculation of the service node when joining or leaving is constant. At the same time, when the new service node joins, the operation amount of each service node is the same, and does not increase with the growing of the service node. It is suitable for the service chain which does not depend on the dedicated hardware facilities and the easy changing network topology.

References