Dear Sir,

Amazon Developer Committee.

I want describe our new approach for Authentication and Group key Generation.

We are in YAS for electronic system, achieve to new model of authentication called “symmetric polynomial authentication for distributed executers” SPADE. This model generate the group key without send through network.

In attachment of email,

Thank you.

Iyad Shaheen

YAS,

UAE.

**Authentication and Group Key Establishment Scheme**

1. **Initialization phase**

In this phase, the manager periodically transmits its certificate to all executers in the system. The certificate is generated using X.509 certificate authentication [Bhatnagar 2015] and it contains this manager’s public key (PKm). As an initialization, every executer, which likes to work with the manager, sends a connect-request, as depicted in (1) to the manager encrypted with this manager’s public key. The connect-request message includes the executer identity, ID, a random number, nonce1, and a timestamp, ts1.

conn − req : EPK m (executerID||nonce1||ts1) (1)

where EPK m () is the encrypted message using the manage public key, PKM, nonce1 is a random number generated by the executer, and ts1 is a timestamp at the executer.

1. **Authentication Phase**

In this phase, the manager authenticates the executer that transmitted a connect-request as the follow steps:

1. Upon receiving the connect request, the manager decrypts the request using its private key PRm. the server then sends a challenge message to the executer, include the received connect-request and another random number, nonce2, and a new time stamp, ts2, as follows:

Chlg: Enonce1 (executerID||nonce1||ts1||nonce2||ts2, F(x, y)) (2)

Where, F(x,y) is the symmetric polynomial that the manager creates for using in creating a group key, L, as depicted in next phase.

1. The executer decrypts the challenge message using its created nonce1 and replies with the response message, Res, as follows:

Res : Enonce2 (executerID||nonce2||ts2) (3)

3) Upon receiving the response message, the manager decrypts the message using nonce2, then compares the values of nonce2 and ts2 at the response and challenge messages and authenticate the executer is the values are identical.

4) If the manager authenticates the executer, it creates a row for this executer in its database include its ID, executerID , and a shared key between the manager and the executer, which is nonce1. Therefore, for subsequent communications between the manager and this executer, the manager uses the share key, nonce1

1. **Group key generation**

After authenticating the executers, the manager chooses group of those executers to be used in encrypting the QRT through VML middleware. First, the manager receives the compressed blocks of the same size, B1, B2, B3,.......Bn. from the DWServer and then transmits it to the chosen executers to encrypt. The manager also works as one of the executers and encrypts a block at its side. The manager needs to send the share encryption key (SKAES) to those chosen executer. However, the key should not be transmitted as clear text because it would be an easy job for the eavesdropper to get the encryption key. Therefore, the chosen executers can create a group key, L with the manager by applying the following steps:

1) After authenticating the executer using the previous phase, the manager chooses a symmetric polynomial F (x, y) and evaluates it for its ID and the executer’s ID, F (managerID, executer 'sID ).

2) If the manager already has a defined symmetric polynomial for another authenticated executer, the manager evaluates this defined polynomial for this new executer.

3) To create the group key, L for n executers, the manager creates the n equations, as depicted in (??), assumes a fixed value for the first modulus, m1,as prime number, then calculates the other modulus, m2,m3,......mn, and the group key, L .

4) The manager then transmits the modular values, m1, m2, ......mn each of which to the intended executer after encrypting it using executer’s shared key, noncei

5) using its modular value mi , each executer can recalculate the group key by evaluating the symmetric polynomial, F (x, y ) received in the response message, as depicted in previous phase, for the manager’s ID and the executer’s ID, then calculate modulus value as in ():

L = F(managerID, executerIDi ) mod mi ,

For i = 1, ........to..n

**As mathematical proof as following :**

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| In Executors Side | | | | | | In Server Manager Side | | | |
|  | C - X | S - Y | F(C1,S) | m1 | L | F | F-L | m | L with m |
| C1 | 1 | 11 | 1452 | 177 | 36 | 1452 | 1416 | 177 | 36 |
| C2 | 2 | 11 | 11132 | 11096 | 36 | 11132 | 11096 | 11096 | 36 |
| C3 | 3 | 11 | 37026 | 36990 | 36 | 37026 | 36990 | 36990 | 36 |
| C4 | 4 | 11 | 87120 | 87084 | 36 | 87120 | 87084 | 87084 | 36 |