Abstract—According to the current key management scheme exist large computation, trusted centre and key update cycle size is difficult to determine problems, we proposed a scheme that is based on ECC fully self-organization Ad hoc key management in this paper. The scheme use ECC instead of RSA for large computation; in order to solve trusted centre, we use fully self-organization; by using a double key update to overcome the regular updating cycle size problem. Computational cost of this program has a small computation safe, reliability and good scalability for large-scale Ad hoc network.

Keywords—Ad hoc network; Elliptic curve Key Management; Self-organization; key update

I. INTRODUCTION

According to the definition [1], wireless Ad hoc network is formed by the mutual cooperation of a group of independent wireless nodes or terminals, independent of fixed infrastructure of self-creation, self-organizing and self-management network. As Ad hoc wireless networks’ features, and it are vulnerable to server denial attacks [2,3,4]. Traditional network of key management scheme usually uses RSA public key algorithm threshold scheme [5,6] based on trusted center scheme [7], but these scheme can not be simply migrated to mobile Ad hoc networks. Because of inherent characteristics of the mobile Ad hoc networks, such as the limited calculation capacity, dynamic topology changes, it is difficult to find the credible CA in such an environment. Moreover if CA is found, CA will become the bottleneck of the whole system. And if the private key of CA is compromised, the entire network will break off. In addition, Ad hoc key update scheme proposed currently is mostly based on regularly updated program of RSA, in which it’s difficult to determine the update cycle. If cycle is too large, the system is easily attacked, while cycle is too small, because of frequent update for the secret key, the efficiency of system will be greatly reduced.

Aiming at the issues raise above, an efficient ECC-based self-organized key management scheme was proposed in this paper. Comparing with RSA in the same security conditions, the scheme has a shorter key length and smaller computational cost [8]. System key in network was accomplished by coordination of each node, through the combination of threshold mechanisms into the system key, synchronously, introducing instantly updated key mechanisms on the basis of regularly updated to solve the problem of how to determine a detailed cycle.

II. ELLIPTIC CURVE CRYPTO-SYSTEM IN FINITE FIELD

To set up cryptography based on elliptic curve, you need problems similar to factoring the product of two prime numbers and discrete logarithm. Consider the equation $Q=kP$, Where $Q$, $P \in E(a,b)$ and $k < P$. It is easy to calculate $Q$ if $k$ and $P$ are given. But it’s difficult to calculate $k$ if $Q$ and $P$ are given. This is the elliptic curve discrete logarithm problem.

Toward the $Z_p$ curve, we use the cubic equation, which set of variables and coefficients in $\textbf{(0,1} \cdots \textbf{P-1)}$ of values, computing modulo $p$ operation.

$y^2 \mod p = (x^3 + ax + b) \mod p$

It can be proved that if $(x^3 + ax + b) \mod p$ has no repeated factors, based on the set $E_p(a,b)$, we can define a limited group of Abel. This is equivalent to the following conditions:

$(4a^3 + 27b^2) \mod p \neq 0 \mod p$

In order to ensure the security of elliptic curve cryptography, you need to know number of the limited abel group midpoint defined on Elliptic curve. In the finite group $E_p(a,b)$, Point range of the number N is:

$p+1-2 \sqrt{p} \leq N \leq p+1+2 \sqrt{p}$

Thus, the number of points in $E_p(a,b)$ is nearly equal to the number of elements in $Z_p$ which is p.

III. IMPLEMENTATION OF THE PROGRAM

A. Security selecting parameters

Assume that n nodes V is the set of members: denoted $V=\{V_1, V_2, \cdots, V_n\}$, System's public parameters are determined by each member of joint consultation, denoted $(p, q, t, E, A, B, H, \{ID_1, ID_2, \cdots, ID_n\})$ Where: p, q for large prime numbers, t threshold for the system; E as the elliptic curve over the finite field $\textbf{Z}_p$.
curve $Z_p$; A, B to E are on the order of two basis points q; H is an open one-way Hash function, ID is the identity of $V$, in which $i \in (1, 2, ..., n)$.

B. System initialization

Step 1: In order to negotiate the group key efficiently for the node in the network, each node in network need to produce its own public private key pair in $t\_0$ Node $V_i \in V$, the randomly selecting initial private key $x_i \in Z_q$, calculate $V_i$ public key $y_i = x_i \cdot B \bmod p$ by using elliptic curve, where $(x_i^n, y_i^n)$ is a point on the elliptic curve defined above. And broadcast $y_i$.

When each node receives the public key broadcasted by other nodes, ensuring that different nodes produce different public key, or one of them will choose its own private key, and calculating $y_{h_{ij}} = x_i \cdot y_i = x_i \cdot x_i \cdot B \bmod p_i$ and broadcasting $y_{h_{ij}}$.

Step 2: Each node $V_i \in V$ randomly selects share of sub-keys $a_{ij} \in Z_q$. Define group of private key is $X = \sum_{i=1}^{n} a_{ij} \bmod q$, and group public key is $Y = X \cdot B \bmod p$.

Step 3: Each node $V_i \in V$ randomly selects coefficient $a_{ij} \in Z_q$, where $r \in (1, 2, ..., k-1)$ K-1 polynomial structure: $f_r(x) = a_{i0} + a_{i1}x + ... + a_{ik-1}x^{k-1} \bmod q$ calculation of the initial validation parameter: $\beta_{i0} = a_{ij} \cdot A \bmod p, \beta_{ij} = \alpha_{ij} \cdot A \bmod p$

$\beta_{i0}, \beta_{ij}$ calculation of the initial secret share:

$p_{ij} = f_r(ID) \bmod q$, $j \in (1, 2, ..., n)$, use public key of $V_i$ to encrypt Key share $p_{ij}$ when t=0, that is: Secret share encryption $\ell_{ij} = ENC_{V_i}(p_{ij})$, use private key $x_i$ of node $V_i$ sign to information when t=0:

\[
SIG_{a_{ij}}(a_{ij-ID}, t_{ij-ID}^{\text{node}(1, 2, ..., n)}) \bmod q
\] (1) broadcast \((1)\) and \((a_{ij-ID}, t_{ij-ID}^{\text{node}(1, 2, ..., n)})\) confidential Information $p_{ij}$.

Step 4: After each node has received n-1 confidential information \((\ell_{ij} = ENC_{V_i}(p_{ij}))\), decrypt information by using $x_i$ and get $p_{ij} = DNC_{x_i}(\ell_{ij})$. To prove the correctness of Secret share of $V_i$, given by $V_i$, we can use the following formula:

\[
p_{ij} = A \cdot \sum_{k=0}^{k-1} a_{ik} \cdot ID^k_i \bmod p
\] (2)

If passed, it is proved the correctness, and if not, $V_i$ cheats $V_i$ and the broadcast fails), calculating $p_{ij} = \sum_{k=1}^{k-1} a_{ik} \cdot ID^k_i \bmod q$. Sub-group public key is $Y_i = p_{ij} \cdot B \bmod p$ and broadcast Sub-group public key $Y_i$, saving $u_{ij}$.

According to the above four steps, we complete the system initialization of each node in order to carry out the following sub-group key update and key-producing work.

IV. DOUBLE THE SHARE OF SUB-KEY UPDATET

Double the share of sub-key update mechanism is set up to prevent attackers break the Group key system or leak share of sub-keys in share of the regular update cycle of sub-keys. Updated on a regular basis or in a timely manner in which the new update members of the node key share on the precondition of ensuring the constant invariability of group key. As for regular update, assume each member node in system used a number of mechanisms to monitor whether there is disclosure of share of sub-keys.

A. Double the share of sub-key update

Step 1: Regularly updated initialization

Each node member $V_i \in V$ in network randomly selects polynomial coefficients $\alpha_{ij}, r \in Z_q$, where $r \in (1, 2, ..., k-1)$ K-1 polynomial structure:

$\varphi_{ij}(x) = \alpha_{ij} \cdot x + \alpha_{i2} \cdot x^2 + ... + \alpha_{ik-1} \cdot x^{k-1}$

and calculate updating verification parameters:

$\delta_{ij} = \alpha_{ij} \cdot r \cdot B \bmod p$

at the same time calculate updating share of sub-keys:

$\Psi_{ij} = H_i(ID) \bmod q, i \in (1, 2, ..., n)$ (3)

Then use the pub key $y_{h_{ij}}$ of members of the node $V_j \in V$ in last cycle to encrypt information and get:

$\tilde{x}_{ij} = ENC_{y_{h_{ij}}}(\Psi_{ij})$.

Use the information of private key $x_{ij}$ in last cycle node member $V_j$ to sign and get:

$SIG_{\tilde{x}_{ij}}(x_{ij} \cdot \{\delta_{ij}, \{H_i(ID), \tilde{x}_{ij}\}_{p_{ij}}\}_{p_{ij}})$ and broadcast3 and the following type 4:

$\{\delta_{ij}, \{H_i(ID), \tilde{x}_{ij}\}_{p_{ij}}\}_{p_{ij}}$ (4)

save the updated share of the sub-key $\Psi_{ij}$.

Step 2: Verification phase

Each member of node $V_i$ in network receives n-1 $\tilde{x}_{ij}$, use $x_{ij}$ to decrypt and get secret share in last cycle: $\Psi_{ij}$, then verify by using the following formula (5):

$\Psi_{ij} = B_r^2 \sum_{r=0}^{k-1} \delta_{ij} \cdot ID_r^k \bmod p$ (5)

If the formula is passed, $\Psi_{ij}$ is correct, and then
calculate the share of node member $V_i$ updating key:

$$\Psi_i = \sum_{j=1}^{n-1} \Psi_{h,j} \mod q$$

If the formula isn’t passed, $\Psi_{h,j}$ isn’t correct and node member $V_i$ broadcasting $\Psi_{h,j}$ to network fails.

Step 3: Update stage
Node member $V_i$ in network needs to update the former polynomial to the new one, as the follow (6) shows:

$$f_x(x) = f_x(x) + q(x) + a_{i,j} + a_{i,j} \cdot x + a_{i,j} \cdot x^2 + a_{i,j} \cdot x^3 \mod q$$

(6)

Authentication parameters updated by calculating are not broadcasted to network, as the follow (7) shows:

$$a_{i,j} = a_{i,j} \cdot A \mod p, a_{i,j,2} = a_{i,j,2} \cdot A \mod p$$

(7)

Calculate Sub-group private key updated by member node $V_i$:

$$Y_{i} = \sum_{j=1}^{n} (V_{h,j} + \Psi_{h,j}) = Y_{h,i} + \Psi_{h,j} \mod q$$

and broadcast sub-group public key share:

$$Y_{h,i} = \Psi_{h,i} \cdot B + Y_{h,i} \mod p$$

and broadcast sub-group public key share to network. Cancel Sub-group public key $Y_{h,i}$ in last cycle.

Each member node in network randomly selects $r_{h,j}$ in $Z_q$, and updates the private key of Members node in cycle $t$: $y_{h,i} = y_{h,i} + r_{h,j} \mod q$, and cancel private key $x_{h,i}$ in last cycle and broadcast updated public key to network:

$$y_{h,i} + r_{h,j} \mod q$$

save other public key of member node $\{y_{h,i}\}_{j=1,2,...,n}$ in network.

B. Sub-key share update in time
We will give a brief introduction about sub-key share update in time, which is the same as the principle sub-key regularly updating is based on.

Case 1: Before network update sub-key for the N time, if you don’t find a sub-key with leak or suspected leak of some node in the monitoring time $t$, you needn’t update key.

Case 2: Before network update sub-key for the N time if you find a sub-key with leak or suspected leak of some node in the monitoring time $t$, you need determine the number of sub-key exposure. If the number is not larger than threshold $t$, you needn’t update key and if it is, you need update key immediately.

V. PERFORMANCE ANALYSIS OF SCHEME
A. Analysis of passive attacks and active attacks
The security of the scheme is base on intractability of elliptic curve discrete logarithm of, any attacker can not get system private key information from the broadcast information; it is impossible for passive attackers get $\beta_{i,j}$ from initial validation parameters $\beta_{i,j} = a_{i,j} \cdot A \mod p$, therefore, any node can not get the system private key $X = \sum_{i=1}^{n} a_{i,j} \mod q$; in the producing process of system public key, any subsystem public key from Honest members can not get subsystem private key so that system private key can’t be obtained; passive attacker can not obtain system private key from public key $Y = X \cdot B \mod p$.

Any active attacker who wants to get $\zeta'_s \neq \zeta_s$ suit for $H(\zeta'_s \cdot A, \zeta'_s \cdot B) = \Phi_s$, according to the above knowledge we can obtain $\zeta'_s$, changing subsystem public key $Y'_s = p'_s \cdot B \mod p$, and last tampering system public key; active attacker obtains $Y'_s \neq Y_s$ with $Y_s$, $\Phi_s$ and $H_s$ to satisfy $\Phi_s = H_s \zeta'_s \cdot A, \zeta'_s \cdot B = H_s \zeta_s \cdot A + \Phi_s \cdot H_s \zeta_s \cdot A + \Phi_s \cdot H_s \zeta_s \cdot A$, according to the nature of Hash Function, it is also impossible, so active attacker can not tamper system public key eventually.

B. Efficiency analysis
According to the above analysis of the safety it is easy to know that our program has a better safety performance. The program is for security needs, when system sub-key share was in consultation, the node's private key and public key generated by the nodes themselves and calculation. Although traffic has increased, it is essential. Because there is no trust in the relationship between the mobile Ad hoc networks, it is unreliable use the key sharing based on the various participants are honest and credible assumptions. Therefore, the program uses Verifiable key sharing to replace basic threshold scheme, actually each node only increases computational overhead in authentication. In addition, in order to prevent leakage of the system private key the program uses a double update mechanism, by which each node increases calculation and communication by a round. In other programs, although distribution of key share by using a trusted CA or identity-based programs TA for the new node is easy, it is useless and unrealized in mobile Ad hoc Networks because response node will disclose their own share of the key and requesting party doesn’t requires authentication to received key share. Our program can select the appropriate update cycle $t$ according to the specific distribution size of the network and other factors, supplementing that update in time effectively solves the problem that update cycle is too large to bring security risks and tool small to reduce the system efficiency, to achieve the best effect.

VI. CONCLUSION
According to the analysis of recent Ad hoc network key management scheme, the paper finds there are problems in it, such as large computational cost, existence of trusted center bottlenecks, update circle of indetermination and so on. This paper proposes an efficient key management scheme for Ad hoc networks, which has small computational cost. Network node arranges system key by through self-organization, efficiently preventing the risk of trusted being attacked. It greatly increases efficiency of
system using a single key on a regular basis update by introducing key update in time, suitable for using in large-scale Ad hoc network.

REFERENCES


