Comparing Symmetric-key and Public-key based Security Schemes in Sensor Networks: A Case Study of User Access Control

Haodong Wang, Bo Sheng, Chiu C. Tan, Qun Li
College of William and Mary
Williamsburg, VA 23187-8795, USA
{wanghd, shengbo, cct, liqun}@cs.wm.edu

Abstract

While symmetric-key schemes are efficient in processing time for sensor networks, they generally require complicated key management, which may introduce large memory and communication overhead. On the contrary, public-key based schemes have simple and clean key management, but cost more computational time. The recent progress of elliptic curve cryptography (ECC) implementation on sensors motivates us to design a public-key scheme and compare its performance with the symmetric-key counterparts. This paper builds the user access control on commercial off-the-shelf sensor devices as a case study to show that the public-key scheme can be more advantageous in terms of the memory usage, message complexity, and security resilience. Meanwhile, our work also provides insights in integrating and designing public-key based security protocols for sensor networks.

1 Introduction

A main challenge of large scale sensor networks is the deployment of a practical and robust security mechanism to mitigate the security risks exposed to the unattended and resource constrained sensor devices. Motivated by the fact of insufficient hardware resources, a great deal of research has focused on the symmetric cryptography based solutions [6, 9, 3, 18] for light-weight computation. These symmetric-key schemes, however, require complicated key management that may cause large memory and communication overhead. This drawback has not yet been investigated by experimental work so it is not clear how these schemes perform in a realistic system.

Recent progress in implementation of elliptic curve cryptography (ECC) on sensors [7, 8] proves public key cryptography (PKC) is now feasible for resource constrained sensors. Given the efficient low-layer primitive in place, the high-layer PKC based security scheme design in sensor networks, however, is not straightforward due to the special hardware characteristics and requirements of sensor networks. Therefore the performance of PKC based security schemes is still not well investigated. This paper compares the symmetric cryptography and PKC based schemes through an experimental study on an important sensor network security problem: user access control. Our results suggest the PKC based user access control scheme be more advantageous in terms of the memory usage, message complexity, and security resilience.

Sensor data access control becomes an important security component as the in-network data storage applications [17] have been proposed for the sensor platforms with cheap and large storage capacity. To protect the data, sensors have to authenticate the user, and control the access to their data. Existing access control scheme on the Internet [10] is not feasible for sensor networks due to the limited power, memory and communication bandwidth. In addition, access control in sensor networks differs from the conventional schemes in that it is not enough to simply deny unauthorized accesses to the data. An unauthorized user should not be allowed to use the network since the network bandwidth is very limited and, more importantly, the battery power of each node may be depleted after malicious users flood messages to the network. Our access control scheme proposed in this paper is composed of three components. First, the sensors in proximity need to exchange pairwise keys for secure communications. Second, the user needs to get authenticated by the local sensors either for local sensor data access or for remote sensor data access. Third, the local sensors also need to help the user and the remote sensor build a pairwise key to achieve end-to-end security.

In summary, we make the following contributions in this paper. (1) We have designed a suite of ECC-based access control protocols including pairwise key sharing between neighboring sensors, local access control, and remote access control. We believe the integral security application sheds new insights into the practicality of the PKC based scheme in sensor networks, and provides a deeper understanding of the security protocol design in a resource constrained system. (2) We provide a detailed comparison of symmetric cryptography and PKC based user access control protocols. Our evaluations are based on actual implementation on commercial off-the-shelf sensor hardware.

2 Related Work

The user authentication and communication encryption have received extensive attentions [10, 11] for security in

large network system. Kerberos [10] has been widely used in distributed client-server authentication and session key establishment. In sensor networks, SPINS protocol [11] shares the same security architecture. While the centralized schemes have many attractive security features, the communication overhead becomes a major issue when the network size scales, specially for the extremely energy constrained sensor nodes in a large network.

A number of key establishment schemes based on predistribution have been explored recently [3, 4, 6, 9]. While these symmetric key based schemes are computationally efficient, the trade-off has to be paid for complicated key predistribution and key management. The public key based pairwise key schemes proposed by Zhang *et al.* [19] achieve some nice security features by using ID-based cryptography. However, the ID-based cryptography is still not feasible for resource constrained sensors.

The most related research to the user access control is [18, 14]. [18] proposes several schemes to restrict and revoke the access privilege of a mobile sink. While this scheme requires a pre-determined moving track for the mobile sink, our scheme addresses a more general user/sensor communication problem. The mobile sink can be regarded as one type of special users in our scheme. Although [14] describes a symmetric-key based local endorsement scheme which is similar to the threshold endorsement in this paper, the symmetric cryptography based protocol suffers larger communication overhead and requires prohibitive amount of memory storage space.

3 System Model and Assumptions

We consider a large scale wireless sensor network deployed in a variety of environments. A user equipped with a portable computing device, such as a PDA, interacts with the sensor network for data query and retrieval. The user can query either "local" sensors through direct communication links, or "remote" sensors (that are outside of direct communication range) through multihop routing by intermediate sensors. We assume an off-line certification authority (CA) that deploys a 160-bit (key size) ECC cryptosystem is responsible for generating all security credentials. The user acquires her certificate from the CA through an out-of-band security channel, which includes an access control list which defines his access privilege.

The adversary may launch either passive attack or active attack, or both. The passive attack includes message eavesdropping and traffic monitoring. For active attack, we mainly focus on following three types. First, sensors can be compromised, but the number of compromised sensors is less than t (where t is a system parameter). The compromised sensor may capture the legitimate user information while being accessed and reveal it to the malicious third party. Second, user collusion can help malicious users to subvert the system and gain more access privilege. We do not bound the number of colluding users. Third, the adversary may flood fake user queries in the network to deplete the battery power of sensor nodes.

In this paper, we do not address disruption attacks. Disruptions occur when the adversary, by compromising a sensor node, drops legitimate messages or contributes a bogus endorsement share in remote access control (as we will describe later) to invalidate user remote queries. While disruption attacks in general are difficult to defend against in sensor networks, such attack is rare since incidents of message dropping and user remote access failure will easily trigger system attentions and expose the compromised sensors.

4 Access Control Schemes

A lot of sensor operations, including the user endorsement in remote access control we will discuss, are achieved through the collaboration of multiple neighboring sensors. To prevent the adversary from eavesdropping, the sensors need to establish pairwise keys with each other and achieve the secure communication channels. Similarly, the pairwise key is also required between the user and the queried sensor for the above reason. In this section, we start the discussion with our certificate based pairwise key establishment scheme. This scheme also can be applied for local access control with slight modification. Finally, we propose a novel remote access control scheme based on threshold endorsement.

4.1 Pairwise-key Establishment and Local Access Control

A common way to share a pairwise key between two parties is to use the Diffie-Hellman (DH) scheme. However, DH is not suitable for sensor networks due to the potential Man-In-The-Middle (MiTM) attack. We develop a certificate-based key establishment scheme adopted from ElGamal encryption [5] over ECC. Instead of exchanging the public key directly as in DH, the sensor derives the public key from the certificate. This public key is then used to generate the challenge. The successful response from the challenged sensor will prove the authenticity. At the same time, the pairwise key can be secretly transmitted by using public key encryption.

Due to the space limit, we do not present the protocol in this paper, the interested readers may refer to a similar scheme described in [14]. This pairwise key establishment scheme requires three ECC point multiplications. Optimizations can be made to reduce the number of point multiplications under certain assumption. For an example, if the sensors have additional storage space like flash memory, pre-computation can reduce one point multiplication. For convenience, we denote "ECC-Cert" and "ECC-PreComp" as the certificate-based key establishing scheme and the optimized scheme with pre-computation, respectively, throughout the rest of the paper.

"ECC-Cert" also can be applied to the user local access control. In that case, the user, say Alice, has to have a certificate C_A attached with her access list al_A (describing the access permission). The queried sensor constructs Alice's public key from C_A and al_A, and then performs the rest of the challenge-response scheme.

4.2 Remote Access Control

Theoretically, a simple extension of the certificate based scheme discussed previously can be used in the remote query. In that case, the challenge-response messages between the user and the remote sensor are routed by a number of intermediate sensors on the relay path. This multihop communication pattern, however, poses new security and efficiency issues: (1) potential DoS attacks; (2) high communication overhead for the user authentication and end-to-end security. The two issues are not found in the local query and can not be addressed by the certificate based scheme due to the following two reasons.

First, because the certificate based access control achieves end-to-end security, any intermediate sensor has no knowledge about the challenge-response message and would not detect the DoS attack had the adversary injected a large number of fake queries.

Second, the message overhead becomes critical in the multihop communication to reduce the energy consumption of intermediate sensors. The certificate based scheme requires public key exchanges between two parities. In practice, the public key size (40B) is larger than the typical message size in sensor networks (29B). This overhead may force the sensor to use multiple data packets to transmit the query that otherwise would be done by just one packet. While the certificate based scheme achieves the user authentication and end-to-end security, it requires two rounds of communications that carry the public keys and incurs the large overhead.

Therefore, we develop a threshold endorsement scheme (inspired by the Shamir's secret sharing [12]) to perform the remote access control. The basic idea is that any user has to be authenticated and endorsed by t local sensors before she can send the remote query. Not only do the t local sensors block any DoS attack attempt and transfer the trust (of the authenticated user) to the remote sensor, given the assumption that the adversary can not compromise t sensors, their endorsements also naturally serve as the pairwise key between the user and the remote sensor without any public key transmission. The three components: DoS prevention, user authentication, and message security are integrated organically in the remote access control scheme.

Our scheme is presented as follows. We have an elliptic curve E over finite field $GF(p)$ and a base point P with the order of a prime q. CA maintains a secret polynomial: $f(y) = 1 + a_1 y + \cdots + a_t y^t$, where $a_i \in GF(q)$ for $1 \leq i \leq t$. Before the deployment, each sensor s_i (s_i denotes the sensor ID) is pre-loaded with a secret share z_i, where $z_i = f(s_i)$. Any $t+1$ shares of secret can reconstruct the secret polynomial by Lagrange interpolation: $f(y) = \sum_{i=1}^{t+1} z_i \prod_{j=1, j\neq i}^{t+1} \frac{s_j - y}{s_j - s_i}$. When $y = 0$, the $t+1$ secret shares satisfy:

$$\sum_{i=1}^{t+1} z_i l_i = 1. \quad (1)$$

l_i is the Lagrange coefficient, and determined as $l_i = \prod_{j=1, j\neq i}^{t+1} \frac{s_j}{s_j - s_i}$.

CA also defines a cryptographic hash function $\overline{H}$, mapping a number $\{0,1\}^*$ to a nonzero elliptic curve point on E. The remote access control protocol is given in Fig. 1. We denote $s_1, s_2, \cdots, s_t$ as the local sensors, s_r as the remote sensor. We assume that the ID of the remote sensor for data access is known by some scheme that is beyond the scope of this paper, e.g., resource discovery protocols.

The user, Alice, first performs local access control protocol with t local sensors, $s_1, \cdots, s_t$. After the successful authentications, each local sensor s_i endorses Alice in the following way. First, s_i calculates $R_A = \overline{H}(al_A)$. Note R_A is a point on the elliptic curve E. Then s_i generates its endorsement: $z_i l_i R_A$, where the Lagrange coefficient $l_i = \prod_{j=1, j\neq i}^{t} \frac{s_j}{s_j - s_i} \cdot \frac{s_r}{s_r - s_i}$ (here we use s_r instead of s_{t+1}). In the next step, s_i sends the endorsement to Alice through the secure communication channel established in the local access control as described in Section 4.1. With the t endorsements collected, Alice calculates the elliptic point V_A, which is the summation of the t endorsements. Note only Alice knows the value of V_A. None of t local sensor knows V_A because each sensor only knows its own share of V_A. Now, V_A becomes the shared secret between Alice and the remote sensor s_r. Alice encrypts her access list and query by $X(V_A)$, the value of X-coordinate of V_A, and then sends the encrypted query along

$Alice \rightarrow s_1, \cdots, s_t : al_A || C_A$
for (each sensor $s_i, i = 1, 2, \cdots, t$)
  s_i : perform user authentication
  s_i : compute $R_A = \overline{H}(al_A)$
  $s_i \rightarrow Alice : z_i l_i R_A$
$Alice$: gets $V_A = \sum_{i=1}^{t} z_i l_i R_A$
$Alice \rightarrow s_r : al_A || l_r || (al_A || query)^+_{X(V_A)}$
s_r : compute $R_A = \overline{H}(al_A), V'_A = R_A - z_r l_r R_A$
$s_r : al_A = ((al_A || query)^+_{X(V_A)})^-_{X(V'_A)}$
$s_r \rightarrow Alice : (reply)^+_{X(V_A)}$

Figure 1. ECC-based local threshold endorsement scheme to establish remote pairwise key between the user and the remote sensor.

with her access list and l_r ($l_r = \prod_{j=1}^{t} \frac{s_j}{s_j - s_r}$, also calculated by Alice) to the remote sensor s_r. Upon the receipt of the remote access request from Alice, s_r first calculates $R_A = \overline{H}(al_A)$ and computes $V'_A = R_A - z_r l_r R_A$. According to Eq.(1), V'_A should be equivalent to V_A because: $\sum_{i=1}^{t} z_i l_i R_A + z_r l_r R_A = (\sum_{i=1}^{t} z_i l_i + z_r l_r) \cdot R_A = R_A$. Therefore, s_r can successfully decrypt al_A and $query$. Finally, s_r replies $Alice$ with the query result, again encrypted by $X(V_A)$.

In summary, the main idea of remote access control is to design a mechanism that allows a set of local sensors (because we do not trust a single sensor) to transfer the trust (if the user is authenticated) to the remote sensor, so that the remote sensor does not need to perform the interactive user authentication employed in local authentication, which requires several rounds of communications. This endorsement scheme can be combined with existing en-route filtering schemes, such as SEF [16], to further prevent the adversary from injecting the data queries through a compromised sensor.

Our scheme can also be extended to work in a sparse network, where t local sensors are difficult to find at one time. In that case, the user moves around and finds t sensors at different locations. To produce the endorsement shares, t sensors need to communicate with each other and exchange their ID list and agree on the remote sensor s_r. Note the communications cannot be initiated by sensor themselves since multi-hop communications have to be endorsed as we described previously. For this reason, the user moves back and force, as a carrier, to distribute the node IDs to each of t sensors. Once t sensors share their IDs and agree on s_r, the rest of scheme is the same as described previously.

Cost and Security Analysis. To endorse the user, each local sensor only needs to perform one ECC point multiplication and one hash function $\overline{H}$. $\overline{H}$ is a special hash function that maps $\{0,1\}^*$ onto the elliptic curve E. According the study by Boneh *et al.* [1], this special hash function can be efficiently achieved by two steps: first we hash onto a certain subset $F \subseteq \{0,1\}^*$;then we use a deterministic encoding function to map F onto E. The message complexity for the threshold local endorsement is small. Each sensor only needs to send an elliptic curve point to the user, which has the message size of 40-bytes (for the 160-bit ECC).

The proposed remote access control scheme is resilient to any sensor compromising attack with no more than $t-1$ compromised sensors due to the property of the threshold cryptography. Each sensor s_i has its own unique secret z_i. Any $t-1$ or less shares of secrets are not enough to recover the secret polynomial [12], and cannot be utilized to deduce the value of z_r hold by the remote sensor.

As described in the protocol, the user knows each share of endorsement: $z_i l_i R_A$, and even $z_r l_r R_A$. Combining all these shares only allows the user to establish shared secret with the remote sensor. These shares can not be used to generate the endorsement for any other access list. Suppose the user has a forged access list al'_A, and the corresponding $R'_A = \overline{H}(al'_A)$. To generate the endorsement shares $z_i R'_A$ ($1 \le i \le t$), the user has to know z_i. However, it is computationally infeasible to retrieve z_i from $z_i l_i R_A$. Meanwhile, the knowledge of $z_i l_i R_A$ cannot be used to derive $z_i l_i R'_A$. The reason is that R_A, R'_A are random elliptic curve points, it is computational infeasible to derive $r_A, r'_A \in GF(q)$, so that $R_A = r_A P$ and $R'_A = r'_A P$. As the result, it is impractical to derive $z_i l_i R'_A$ from $z_i l_i R_A$. For the same reason, the user cannot reuse the acquired secret endorsement to access a different remote sensor.

Since each endorsing sensor establishes a secure communication channel with the user during the local authentication, the adversary cannot capture any share of the endorsement by eavesdropping. Therefore, only the user and the remote sensor share the secret, which is to build the secure communication channel for the remote access.

5 Analysis and Evaluation

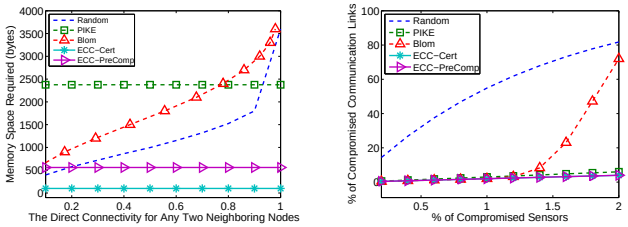
We evaluate our access control schemes by a combination of theoretical analysis and implementation on a sensor platform. We first conduct theoretical analysis to compare our key establishment scheme with Random-key [6], PIKE [3], and Blom [4]. We then implement Random-key [6] and Blundo [18] on MICAz sensors to perform experimental comparison with our schemes. Finally, we implement all components in the proposed remote access control. By focusing on the processing delay, we demonstrate the delay is small, which makes our scheme practical in the real world.

5.1 Analytical Results

The metrics used to compare pairwise key establishment are **memory overhead** and **security resilience**. The symmetric key schemes require key pre-distribution, the memory overhead measures the amount of memory space required to achieve a certain degree of key connectivity between two nodes. In security resilience against the node compromise, we measure the fraction of the compromised communication links as a result of sensor compromise.

For key establishment, we compare aforementioned symmetric key schemes with our two variations of our ECC-based pairwise key schemes: ECC-Cert and ECC-PreComp, which were discussed in Section 4.1. Our analysis is based on a randomly, uniformly deployed sensor network with 10,000 nodes. On average, each sensor has 10 neighbors. The senor node IDs have the size of 2 bytes. The random keys have the size of 10 bytes. With additional 2 bytes for key indices, each pre-distributed random key requires 12 bytes for memory space. We assume the key pool size is 10,000 for both Random-key and PIKE. We choose 160-bit ECC as our public key primitive. Therefore, the

ECC certificate has 40 bytes, each ECC public key has 40 bytes, and ECC private key has 20 bytes.



(a) (b)

Figure 2. (a). The memory space required for any two nodes to establish a direct pair-wise key under different key connectivity rate. (b). The trend of percentage of total communication links compromised with the increasing number of sensors compromised.

The ability to establish a direct pairwise key (not through the third party) between two neighboring sensors is very important, since direct key sharing not only reduces the communication overhead, more importantly, also improves the security resilience. Fig. 2(a) shows the memory overhead required by the key establishing schemes to achieve a direct key between two sensors with different probability.

To increase the probability of establishing direct pairwise key, Random-key scheme needs to pre-distribute more keys in each sensor node. We can see from Fig. 2(a), the memory overhead is increasing linearly when the required key connectivity increases from 0.1 to 0.9. This trend becomes exponential when the connectivity is larger than 0.9. To achieve 100% connectivity, each sensor has to be preloaded with 300 keys, which requires 3.6KB memory space. Considering that MICAz only has 4KB data space, the 300 keys almost consume all available memory and leave almost no space for the application programs. Thus, the Random-key scheme obviously is not practical to achieve 100% direct key connectivity.

The memory overhead of PIKE only depends on the network size. Given 10,000 sensor nodes, each sensor has to be pre-loaded with $2 \times (\sqrt{10000} - 1) = 198$ keys. Therefore, the memory overhead for PIKE is constantly $12 \times 198 = 2,376$ bytes. Blom scheme with $\lambda = 29$ and $\omega = 50$ (please refer [4] for the details) also introduces high memory overhead, specially when the high key connectivity rate is required.

Compared to symmetric key schemes, our ECC based schemes have much less memory overhead. In ECC-Cert, each sensor has to store its public key and private key, as well as the certificate, so the memory overhead becomes 100 bytes. ECC-PreComp has more memory overhead because each sensor needs to store the pre-computed random numbers (20 bytes each) and corresponding elliptic curve points (40 bytes each). Given average 10 neighbors, each sensor stores 10 pre-computed values, which account for 600 bytes more overhead. As a result, the memory overhead for ECC-PreComp is 660 bytes. Note the memory overhead of the public key base schemes does not change for achieving different key connectivity.

Fig. 2(b) shows the fraction of compromised communication links due to the node compromise. Here, the fraction of the compromised communication links include the direct links connected to the compromised nodes and indirect communication links due to the leakage of the system secret, such as the subset of system key pool in Random-key scheme. When the percentage of the compromised nodes is small, we may consider the compromised nodes scattered in different neighborhoods by approximation.

Our ECC based public key scheme is ideal under such situation. There is no indirect link compromised due to the node compromise. For every p (percent) of sensors have been compromised, approximately $2p$ of direct communication links are no longer secure. We denote N, d, e to be the total number nodes, average degree and the number of edges, respectively. Graph theory tells us that $2e = Nd$. If p percentage of nodes are compromised (p is small), pNd direct links are affected. The percentage of these affected link is $\frac{pNd}{e} = \frac{pNd}{Nd/2} = 2p$. In PIKE, since each sensor servers the intermediary for other two sensors, approximately $3p$ of communication links are compromised.

In Random-key scheme, as shown in Fig. 2(b), the number of compromised indirect links dominates the ones directly connected to the compromised nodes. As a result, the fraction of the compromised communication links is much higher than those of ECC-based schemes. When only 2% of nodes are compromised, over 80% of links are compromised. Therefore, the Random-key scheme has much poorer security resilience compared to the ECC-based pairwise key schemes.

The performance of Blom scheme is shown in Fig. 2(b), for both direct links and indirect links. Even though Blom scheme has almost the same security resilience when the number of compromised nodes is small, the security resilience degrades exponentially when more nodes are compromised. When 2% of nodes are compromised, over 70% of links are compromised.

5.2 Experimental Results

We implement Random-key scheme and Blundo user access control scheme as the real world comparison. We use the following four metrics: **key establishing time**, **memory overhead**, **message complexity** and **energy consumption**. The key establishing time measures the time duration for a random sensor to establish secret pairwise key with its neighbors. Similarly, the memory overhead measures the exact amount of data space required (in the real implementation) in the access control. The message complexity then shows the amount of messages transmitted during the key establishing procedure. The energy consumption esti-

mates the average communication energy consumed during the key establishment.

5.2.1 Experiment Testbed and Parameter Setting

We implement our ECC-based schemes on MICAz motes and HP iPAQ. MICAz is powered by a ATmega128 microcontroller, which features an 8MHz, 8-bit RISC CPU, 128K bytes flash memory (ROM) and 4K RAM. The MICAz runs TinyOS [13]. The HP iPAQ is used for the user module. Due to the space limit, we omit the discussion of ECC implementation on MICAz and iPAQ. Interested readers may refer to [15] for details.

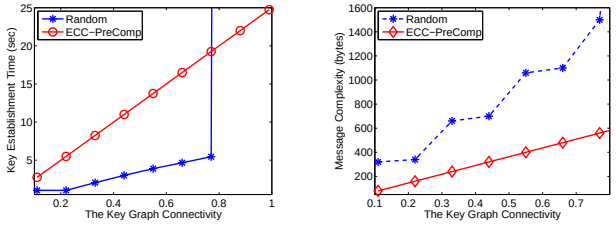
We implement the baseline symmetric key scheme, Random-key, on the same testbed for the comparison of pairwise key establishment. We use 10 MICAz motes to form a sensor neighborhood. Each sensor can directly communicate with any of other nine neighbors. We select the key pool size of 10,000. Each key, with the size of 10 bytes, is identified by a two-byte key index. Each sensor is randomly pre-distributed with 150 keys. In the experiment, we randomly pick one out of ten motes to initiate the pairwise key establishment with all its neighbors. Even with 150 keys pre-loaded, they are not enough for any mote to establish direct pairwise key with all the neighbors. Therefore, multiple rounds of key establishment have to be performed. We limit it to three rounds. That means any two neighboring motes at most have two helpers for establishing indirect pairwise key. This arrangement is supported by the fact indicated in Random-key [6] that the number of pairwise key established through more than 3 hops is negligible.

To implement the Blundo access control, we first generate a random symmetric polynomial. The coefficients have the size of 10 bytes. The polynomial degree is adjustable for the target security resilience against the node compromise. Each mote is pre-distributed with a secret polynomial share, which is generated by simply plugging in the mote ID. The amount of memory space for storing the polynomial share is determined by the polynomial degree. Similar as the access control testbed implemented by our ECC public scheme, we use the HP iPAQ as the user module.

For all schemes conducted on our testbed, we repeat the tests for 20 times, and record the average values.

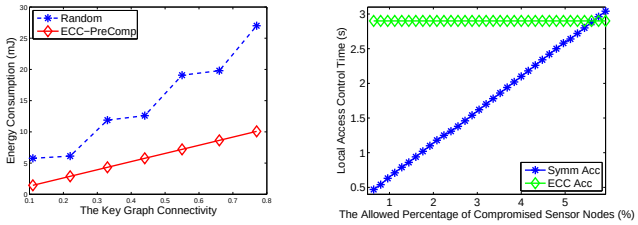
5.2.2 Pairwise Key Establishment

Fig. 3(a) illustrates the processing time delay in pairwise key establishing for achieving different degree of key connection. We select ECC-PreComp for this experiment. Compared to ECC-PreComp, Random-key scheme has lower processing overhead when the requirement of key connectivity is low. However, this advantage does not hold if more than 80% key connection is required. The reason is that the number of pre-distributed keys is not enough for establishing pairwise keys with all its neighbors. The key establishing time thus increases to infinity. The time jumps to infinite large at key connectivity of 0.8. We restrict the pre-distributed key number due to the limited 4KB memory space in MICAz motes. In our experiment, with 150 key pre-distributed, a mote can only establish direct pairwise key with two out of its nine neighbors. The other pairwise keys are established through the second and the third rounds of key establishing procedure.



(a) (b)

Figure 3. (a). Key establishing delay for different key graph connectivities. (b). The message complexity for achieving the target key connectivity.



(a) (b)

Figure 4. (a). The communication energy consumption for achieving the target key connectivity. (b). Local authentication time vs. security resilience (% of compromised sensors).

Fig. 3(b) further reveals that ECC-based pairwise key schemes have much less message complexity than Random-key scheme. To establish a pairwise key, each sensor only needs to transmit 120 bytes for messages. In Random-key scheme, the broadcasting node has to send all key IDs in its key ring. Given 150 keys and 2 bytes each for key index, the broadcasting mote transmits 300 byte message. All listening neighbors also need to respond the key establishing broadcast, by either replying the challenging message (if there is a shared key), or notifying there is no shared key. This message overhead has to be paid in all three key establishing rounds. In the wireless medium, high message complexity increases the chance of message collision and thus causes network congestion. The low message complexity is a significant advantage for ECC-based pairwise key establishing schemes.

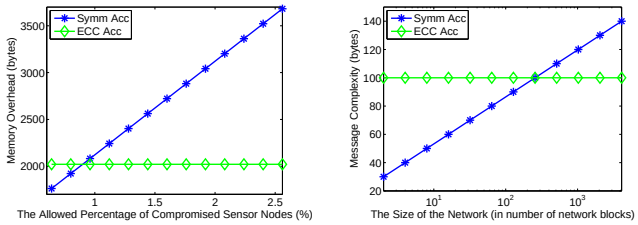
Finally, we compare the communication energy consumption during the pairwise key establishment. We estimate the energy consumption by multiplying the total amount of communications by an average communication energy consumption of $18\mu J$/bit [2]. Fig. 4(a) identifies the key drawback of Random-key scheme. The symmetric key

based scheme consumes much more communication energy than the ECC based scheme. The reason is that message broadcast is required in Random-key scheme. As a result, all neighboring sensors need to listen the broadcasts all the time and consume the energy for receiving the messages.

5.2.3 Local Access Control

We first measure the authentication delay in the local access control. The user authentication delay is shown in Fig. 4(b). When the security resilience is low, up to 5.5% percent of sensor nodes allowed to be compromised, the Blundo access control is more efficient than our ECC-based scheme. The reason is that the polynomial operations are much faster than ECC exponentiation. However, the processing overhead of the Blundo based scheme increases as the requirement of security resilient increases. When the requirement of security resilience is more than 5.5%, the processing overhead of the symmetric key based scheme becomes slower than our public key based scheme. The reason is that the processing overhead of our ECC based scheme does not change, it always provides the security equivalent to the discrete logarithm problem.

Note, the security concern of user collusion attack has not been revealed yet by this experiment. This security issue has to be considered in real world deployment. Therefore, either higher degree random polynomial or multiple polynomial have to be selected to improve the security. As a result, the processing overhead of the Blundo based access control will be higher. On the contrary, the ECC based access control scheme does not suffer from user collusion attack, so our scheme can be directly applied to the real world deployment. Fig. 5(a) shows the comparison of data size



(a) (b)

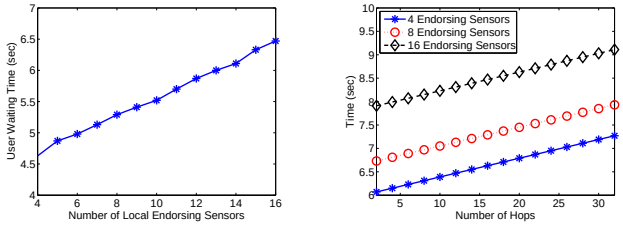
Figure 5. (a). The memory space required to finish the local user authentication. (b). The message complexity in user authentication.

of two local access control schemes. It clearly shows that the memory overhead scales linearly in the Blundo based scheme for satisfying different security resilience. The degree of the random polynomial is larger for higher security requirements. As a result, the sensors need more space to store the corresponding coefficients. The data size of the ECC based scheme, as can be easily predicted, does not change at all.

When Fig. 4(b) and Fig. 5(a) show that the Blundo access control has poor security scalability in processing time and memory overhead, Fig. 5(b) displays that it also has poor network scalability in message complexity. Since the Blundo access control scheme uses "Cell Merging" and "Block Compression"[18] to reduce the number of polynomial possessed by the user. The user has to traverse a Merkle-hash tree. The traversal path length is determined by the tree size, which is in turn determined by the number of location blocks, or the network size. Again, our ECC based user access control has the advantage of excellent network scalability; the message complexity is independent to the network size, fixed at 100 bytes. The figure clearly shows that the Blundo based scheme has more complexity than our public key scheme when the network size is just over 100 blocks. This fact proves our scheme is more favorable for large network deployment.

5.2.4 Remote Access Control

We first provide the microbenchmark for the local authenticate and threshold endorsement generation. Then, we provide the overall estimation of the remote access performance. In the experiments, we mainly focus on the user perceived remote access processing delay. Our first hand experimental results suggest the public key based remote access control scheme is very practical.



(a) (b)

Figure 6. (a). Key establishing time with the remote sensor when the number of endorsing sensor changes from 4 to 16. (b) Remote query time delay.

The local endorsement procedure can be further divided into user local authentication and endorsement generation. We have already demonstrated the performance of user local authentication in the previous section. To be authenticated by multiple local sensors, a simple and effective optimization can be applied to allow the user to be authenticated in parallel rather than one-by-one. The user first sends its certificate to all the endorsing sensors, so that the endorsing sensor can verify the certificate and generate the challenges simultaneously. Then the user collects all the challenges from each member of endorsing group and responds them one-by-one. This optimization is valid because the user device is much more powerful than sensors.

After finishing the user authentication, the local sensors perform threshold endorsement for the user. We continue the above authentication experiment. Each endorsing sensor immediately computes its endorsement share and then sends to the user sequentially. Fig. 6(a) shows the user

waiting time to receive all the endorsement shares. With the number of endorsing sensors changing from 4 to 16, the time duration linearly grows from 4.5s to 6.6s. The measurement includes the user local authentication time (local pairwise key establishing time).

Upon the receipt of the remote access query, the remote sensor has to verify the authenticity of the remote query by decrypting the message using its own secret share as presented in section 4.2. The computational complexity of this operation is independent to the number of local endorsing sensors. The only expensive operation at the remote sensor is one ECC point multiplication. It takes 1.4s for the remote sensor to calculate its secret share and verify the query.

Finally, we investigate the overall performance of the remote access control, including the threshold signature generation, message propagation, and remote sensor verification. We assume the local endorsing sensors have already established pairwise key with each others. To simplify the experiment, the user directly sends the query to the remote sensor. Then we add the estimated hop-by-hop forwarding delay to estimate the performance for various hop distances. The estimated forwarding delay is the communication delay in sensor RF transceiver. Our estimation fixes the amount of communication delay to 17.5ms. [1]

Fig. 6(b) shows the estimated overall user remote query response time, given the size of local endorsing group with 4, 8 and 16, respectively. We find the overall remote query delay is short. When the remote sensor is located at 20 hops away, the user query response time is 6.8s. When the larger size of the local endorsing sensor group is required, the additional overhead increases moderately.

6 Conclusion

This paper proposes an ECC-based access control for sensor networks, which consists of pairwise key establishment, local access control, and remote access control. We have performed a comparison test by implementing both symmetric-key and public-key primitives on popular sensor motes. Our experiment results suggest public key based protocol is more advantageous than the symmetric key in terms of the memory usage, message complexity, and security resilience. To the best of our knowledge, this is the first comprehensive experimental comparison between symmetric-key and public-key scheme in sensor networks.

Acknowledgments

The authors would like to thank all the reviewers for their helpful comments. This project was supported in part by US National Science Foundation grants CCF-0514985, CNS-0721443, and CAREER Award CNS-0747108.

[1] Based on our experimental result of forwarding a 60 byte payload in MICAz motes.

References

[1] D. Boneh and M. Franlin. Identity-Based Encryption from the Weil Pairing. In *CRYPTO*, pages 213–229, Springer-Verlag, 2001.

[2] D. Carman, B. Matt, P. Kruus, D. Balenson, and D. Branstad. Key Management in Ditributed Sensor Networks. In *DARPA Sensor IT Workshop*, 2000.

[3] H. Chan and A. Perrig. PIKE: Peer Intermediaries for Key Establishment in Sensor Networks. In *INFOCOM*, Miami, FL, March 2005.

[4] W. Du and J. Deng. A Pairwise Key Pre-distribution Scheme for Wireless Sensor Networks. In *ACM CCS*, 2003.

[5] T. ElGamal. A public key cryptosystem and a signature scheme based on discrete logarithms. In *In Advances in Crytography - Proceedings of CRYPTO*, pages 211–222, 1985.

[6] L. Eschenauer and V.D. Gligor. A Key-management Scheme for Distributed Sensor Networks. In *ACM CCS*, November 2002.

[7] N. Gura, A. Patel, A. Wander, H. Eberle, and S.C. Shantz. Comparing Elliptic Curve Cryptography and RSA on 8-bit CPUs. In *CHES*, Cambridge, MA, Aug 2004.

[8] An Liu and Peng Ning. *http://discovery.csc.ncsu.edu/software/TinyECC/*, 2005.

[9] D. Liu and P. Ning. Establishing Pairwise Keys in Distributed Sensor Networks. In *ACM CCS*, Washington, DC, October 2003.

[10] B. Clifford Neuman and Theodore Ts'o. Kerberos: An Authentication Service for Computer Networks. *IEEE Communications*, 32(9):33–38, September 1994.

[11] A. Perrig, R. Szewczyk, V. Wen, D. Culler, and D. Tygar. SPINS: Security Protocols for Sensor Networks. *ACM/Kluwer Wireless Networks Journal (WINET)*, September 2002.

[12] A. Shamir. How to Share a Secret. *Communications of the ACM*, 22(11):612–613, 1979.

[13] TinyOS. TinyOS 1.1.15. *http://www.tinyos.net*, 2006.

[14] H. Wang and Q. Li. Distributed user access control in sensor networks. In *IEEE International Conference on Distributed Computing in Sensor Systems (DCOSS)*, pages 305–320, San Francisco, CA, June 2006.

[15] H. Wang, B. Sheng, C. C. Tan, and Q. Li. WM-ECC: an Elliptic Curve Cryptography Suite on Sensor Motes. Technical Report WM-CS-2007-11, College of William and Mary, Computer Science, Williamsburg, VA, 2007.

[16] F. Ye, H. Luo, S. Lu, and L. Zhang. Statistical En-Route Filtering of Injected False Data in Sensor Networks. In *INFOCOM*, 2004.

[17] D. Zeinalipour-Yazti, S. Lin, V. Kalogeraki, D. Gunopulos, and W. A. Najjar. MicroHash: An Efficient Index Structure for Flash-Based Sensor Devices. In *FAST*, 2005.

[18] W. Zhang, H. Song, S. Zhu, and G. Cao. Least Privilege and Privilege Deprivation: Towards Tolerating Mobile Sink Compromises in Wireless Sensor Networks. In *MOBIHOC*, Chicago, IL, May 2005.

[19] Y. Zhang, W. Liu, W. Lou, and Y. Fang. Location-based Compromise-tolerant Security Mechanisms for Wireless Sensor Networks. *IEEE Journal on Selected Areas in Communications*, 24(2):247–260, Feb 2006.