Wireless Netw
DOI 10.1007/s11276-009-0184-z

Achieving robust message authentication in sensor networks: a public-key based approach

Haodong Wang · Qun Li

Abstract Given the extremely limited hardware resources on sensor nodes and the inclement deploying environment, the adversary Denial-of-Service (DoS) attack becomes a serious security threat toward wireless sensor networks. Without adequate defense mechanism, the adversary can simply inundate the network by flooding the bogus data packets, and paralyze the partial or whole sensor network by depleting node battery power. Prior work on false packet filtering in sensor networks are mostly based on symmetric key schemes, with the concern that the public key operations are too expensive for the resource constrained sensors. Recent progress in public key implementations on sensors, however, has shown that public key is already feasible for sensors. In this paper, we present PDF, a Public-key based false Data Filtering scheme that leverages Shamir's threshold cryptography and Elliptic Curve Cryptography (ECC), and effectively rejects 100% of false data packets. We evaluate PDF by real world implementation on MICAz motes. Our experiment results support the conclusion that PDF is practical for real world sensor deployment.

Keywords Sensor networks · Denial of service · Authentication · ECC

1 Introduction

The repertoire of sensor network applications requires an inclement and human unattended environment, such as battlefield surveillance, wild animal habitat monitoring, and environmental monitoring. Given the extremely constrained hardware resources of the sensor nodes, the adversary Denial-of-Service (DoS) attack becomes a serious security threat. The adversary can first compromise an individual low-power sensor, and then inundate the whole network by injecting large amounts of bogus data packets into the network through the compromised node. These bogus messages flood the network, deplete the battery power of the sensor nodes, and finally paralyze the whole network.

This problem has attracted many attentions in the past several years. Most of prior work [1–4], except [5], on sensor network message authentication and bogus data filtering mainly rely on symmetric key schemes. Ye et al. [1, 2] proposed a statistical en-route false report filtering scheme (SEF). The scheme requires each report be endorsed by multiple sensor nodes by encrypting the report with their random pre-distributed symmetric keys. The intermediate nodes on the route compare their own keys with those used for encrypting the report, and check the corresponding encryption if matched keys are found. Since the authentication capability of the intermediate nodes depends on the probabilistic key sharing, only a portion of bogus messages can be detected and dropped. If the communication is between two remote sensor nodes, the receiver still cannot know, with a certain probability, whether or not the message is valid. Zhu et al. [4] proposed an Interleaved Hop-by-hop Authentication scheme (IHA) to detect the false report. The protocol requires that the sensor nodes maintain a pre-route interleaved associations so that any sensor shares each secret with its upper associated node and lower associated sensor. The problem of this approach is that it is not practical for large sensor networks. Many times, the message routing paths are not determined due to the unpredictable nature of wireless

H. Wang (✉) · Q. Li
Department of Computer Science, College of William and Mary, Williamsburg, VA 23187, USA
e-mail: wanghd@cs.wm.edu

Q. Li
e-mail: liqun@cs.wm.edu

Published online: 20 May 2009

communications. The association requires global knowledge of the networks, which is very difficult to get for large scale sensor networks. Further, this scheme only filters the false report which is sent to the sink. The sensor nodes have no ability to authenticate the messages between the sensor nodes since the corresponding association knowledge is not available.

Unlike the symmetric key based schemes, the public key approach [5] proposed by Zhang et al. yields better security resilience. Unfortunately, the bilinear pairing based scheme is too expensive to be afforded by the low-power sensor hardware. Another straightforward public-key based approach is to use the public-key infrastructure (PKI) that is widely used on Internet, e.g., X509. However, PKI cannot be directly used on sensor networks due to following three issues. First, public key size is normally large, such as 128 bytes for 1024-bit RSA. Sensors are extremely resource constrained devices. The distribution of public keys in sensor network would cause high communication overhead, which in turn will reduce the battery life. Second, the public key has to be certified before it can be used to verify a signature. It is difficult to have an on-line CA in sensor networks. The workaround solution is to attach a certificate with the public key. But again it would cause more communication overhead since the certificate has the same data length as the public key. Third, the simple scheme is not resilient to defend against DoS attacks. If a sensor is compromised, the adversary then uses it to send a large number of messages with legitimate signatures (of the compromised sensor).

In this paper, we propose a Public-key based false Data Filtering scheme (PDF), which leverages threshold cryptography and Elliptic Curve Cryptography (ECC). As we will show, ECC is more affordable than other public key schemes for sensors. With carefully devised ECC-based security protocols and optimized ECC primitive implementation on sensor nodes, ECC is very practical on extremely resource constrained devices. In PDF, any event report message requires an attached digital signature which is signed by system private key. Due to the threat of node compromise, any single sensor cannot be trusted to keep the system private key and be allowed to generate the system signature. Instead, with the assumption that the adversary can not compromise up to t sensors, we design a threshold endorsement scheme. We first pre-distribute a unique system secret share to every individual sensor during the network deployment. Upon the detection of an event, the group of sensor nodes that detect the event collaborate together and jointly generate a system signature. The intermediate sensor nodes can easily validate the event report by efficiently verifying the attached signature. Unlike the symmetric key based schemes that only support false data filtering for the sink bounded messages, PDF supports any point to point communication in the sensor network.

Since it is computationally infeasible for the adversary to forge a digital signature without knowing the system secret, any false report will be detected with 100% probability. PDF is also resilient to sensor compromising attack. The threshold cryptography guarantees the system secret will not be revealed as long as no more than $t - 1$ (t is a system parameter) sensors are compromised. We have implemented all the components for the false data filtering scheme on the real world sensor nodes and shown the performance of the public-key based scheme is practical.

The contribution of this paper can be summarized as follows. First, we propose a public-key based false data filtering scheme for sensor networks. Different from symmetric key based schemes, our scheme is able to filter out the false data with 100% probability and support any end-to-end communication in sensor networks. Second, we carefully design a threshold signature generation scheme that allows a number of low-power and untrusted sensors to cooperatively and efficiently generate a system digital signature. Our threshold signature generation scheme can also be applied in other applications, such user access control. Third, we have implemented all the components for our proposed scheme, including the ECC public-key primitive suite for MICAz sensor motes. Our experiment results prove that PDF is practical for real world applications.

2 Related work

Sensor network security has attracted extensive attentions in recent years. Eschenauer and Gligor propose a random graph based key pre-distribution scheme [6]. The scheme assigns each sensor a random subset of keys from a large key pool, and allows any two nodes to find one common key with a certain probability and use that key as their shared symmetric key. Based on their contribution, a number of researches [7–11] have been delivered to strengthen the security and improve the efficiency. Researchers found the sensor deployment information can be used to reduce the number of pre-loaded keys and meanwhile improve the key connectivity. Instead of pre-distributing random keys, schemes [9, 10] pre-loads either secret matrices or secret polynomials in the sensors to improve the connectivity and reduce the overhead. Recently, this method is also adopted in heterogeneous sensor networks [12, 13]. Although the symmetric key based schemes are efficient in computation, they all require considerable memory space and communication overhead for key pre-distribution and key discovery. The public key based pair-wise key schemes proposed by Zhang et al. [5, 14] achieve some nice security features by using ID-based cryptography. Unfortunately, it is still a

doubt that the ID-based cryptography is feasible for resource constrained sensors.

The most related research to our work are [1–4]. Zhu et al. [4] proposed an Interleaved Hop-by-hop Authentication scheme (IHA) to detect the false report. The protocol requires that the sensor nodes maintain a pre-route interleaved associations so that any sensor shares each secret with its upper associated node and lower associated sensor. Ye et al. [2] proposed a statistical en-route false report filtering scheme (SEF). The scheme requires each report be endorsed by multiple sensor nodes by encrypting the report with their random pre-distributed symmetric keys. The intermediate nodes on the route compare their own keys with those used for encrypting the report, and check the corresponding encryption if matched keys are found. If the corresponding encryption does not match, the report is considered as a forged one and dropped. A more sophisticated en-route false report filtering scheme is proposed by Yang et al. [1]. Based on SEF, this scheme pre-distributes the symmetric keys in a way that the keys are associated with the sensor location (in the granularity of a cell). This scheme is more resilient than SEF because the adversary has to compromise a number of sensors in a same location to forge an event report, which is considered more difficult and easier to be detected. Besides the above symmetric key based schemes, Zhang et al. propose a Probabilistic En-route Filtering scheme [5] by threshold-endorsement using ID-based cryptography. The scheme is more resilient and more effective than the symmetric key schemes in defending against the various security attacks, such as Sybil and node duplicates. However, as we mentioned, the computation is still too expensive for practical implementation on real world sensor networks.

The threshold cryptography adopted in this paper was also studied for ad hoc network security in prior work [15, 16]. The main difference is that sensor nodes have extremely limited resources, including CPU, memory, storage and battery power. As a result, their solutions cannot be practically applied to sensor networks. We are the first to implement threshold cryptography in practice for sensor networks. We carefully design the protocol to show threshold cryptography is affordable and practical for real sensor network deployment.

3 Network and security model

We consider a large scale wireless sensor network deployed in a variety of environments. Sensor nodes are the low-cost wireless devices and have very limited hardware resources including processor, memory and energy. Upon detection of an event, the sensor nodes generate event report packets and send them back to the sink through multihop routing. For the event detection that needs the collaboration of a group of nearby sensors, we assume the sensor clustering protocol, as proposed in prior work [17–23], has been already deployed. The event report is generated by the sensor cluster and transmitted to the sink by the multi-hop routing protocol. We assume the sensor network routing scheme, such as Directed Diffusion [24], LEACH [22] or GPSR [25], is also deployed.

The sensor network security is managed by a Certification Authority (CA), which is responsible for generating all security credentials and distributing the secret keys. Due to the constrained resources and costly wireless communications on sensors, the CA can not be online and accessible as the way it runs in Public Key Infrastructure (PKI). Instead, the CA only runs during the network deployment, system rekeying, or sensor replenishing period. Since the CA has to be off-line in most of time, each sensor has to be pre-loaded with its private key, public key and certificate before the deployment. Each sensor uses these keys to build the secure communication channels with its neighboring sensors as well as perform future sensing tasks.

An adversary is assumed to use all possible means to attack the message authentication mechanism in the sensor network. To capture the system secret, the adversary may launch either passive or active attacks. A typical passive attack is message eavesdropping. The active attacks, however, may include Man-In-The-Middle (MITM) and sensor compromise. Due to the limited hardware resources, sensor nodes may be compromised upon capture. In this paper, we assume the adversary can retrieve all secret information from compromised sensors. However, we assume that at most $t - 1$ sensors can be compromised. The assumption is reasonable because compromising sensors takes time and effort. In addition to the system secret capture, this paper focuses on the adversary DoS attack. The adversary may forge the event reports and inundate these messages in the network in order to deplete the batter power of sensors and finally paralyze the network.

Finally, this paper assumes the event detected by a sensor group (or cluster) with t members is always genuine. It is true that the adversary may generate a fake event or a forged value to confuse the base station. The adversary can influence the group decision through various attacks, such as the Sybil attack. However, it is obviously out of the scope of the security problem addressed in this paper, and prior work [26, 27] has already proposed schemes to defend against such attacks. We thus do not explicitly address the security problem in event detection.

4 Public-key based false data filtering (PDF)

In this section, we present PDF, a public-key based false data filtering scheme. The basic idea is to generate a system signature for each event report so that any intermediate node with the system public key can easily verify the event report and drop the false data packets. While public key signature generation and verification have been well established in Internet, its application in wireless sensor network poses a unique challenge. To generate a system signature, the sensor node has to have the system private key. However, any single sensor cannot be trusted to hold the secret because it is vulnerable to adversary's compromise attack. Our PDF solves the problem by using Shamir's secret sharing. Instead of giving the system secret to each individual sensor, PDF distributes the secret in the following way: each sensor holds a unique share of the secret and any t sensor can collaborate together and reconstruct the secret. Therefore, each event report message has to be endorsed by t sensor nodes. The t endorsing sensors actually jointly generate a system signature for the endorsed packet.

We first briefly introduce Shamir's secret sharing scheme. Second, to achieve the least overhead as possible, we then adopt the ECPVS signature scheme [28]. Third, we present the threshold endorsement false report filtering scheme. Finally, we provide the cost and security analysis, as well as the extension of probabilistic verification to reduce the computation cost.

4.1 Shamir's secret sharing

We assume CA maintains a system secret polynomial:

$$f(y) = a_0 + a_1 y + a_2 y^2 + \cdots + a_{t-1} y^{t-1}. \quad (1)$$

$a_0, a_1, \ldots a_{t-1}$ are random number picked in $GF(q)$. System secret x is picked as $x = a_0$.

During the sensor network deployment, each sensor (identified by s_i) is pre-distributed with a secret share of x. In particular, the secret share for sensor s_i is $x_i = f(s_i)$. Any t sensor nodes can reconstruct the system secret by Lagrange interpolation: $x = \sum_{i=1}^{t} l_i x_i$, where $l_i = \prod_{j=1, j\neq i}^{t} \frac{s_j}{s_j - s_i}$ is Lagrange coefficient. However, it is computationally infeasible for any $t-1$ or less sensors to reconstruct the system secret.

4.2 ECPVS signature scheme

The typical digital signature scheme in ECC is the elliptic curve version of Digital Signature Algorithm (DSA), also know as ECDSA. ECDSA produces 40 byte signature, which is much smaller than 128 byte signature of RSA. However, we are still concerned that the 60-byte message payload (combining a 20-byte message and its 40-byte ECDSA signature) is still too large for a typical data packet for sensor network (e.g., 29 bytes in TinyOS for MICAz motes). Therefore, we adopt ECPVS signature scheme which offers smaller signature size than ECDSA.

We describe the ECPVS [28] signature scheme as following. Given a message M, we divide M to $C||V$, where C and V are two parts of the message M, and $|C| + |V| \geq |M|$, because it is necessary to arrange some redundant information to be included in C. For example, C holds some secret information and the signer identity, while V holds the sender identity, message description, time stamp, etc. We assume the signer has her private key x, and the corresponding public key $Q = xP$. The signer performs the following steps to sign the message.

1. Choose a random key k in $[1, q-1]$;
2. Compute kP, resulting a point with coordinate (x_k, y_k), let $r = x_k$. Check $r \pmod q$, go back to the first step if the result is zero;
3. Compute $e = ENC(r, C)$;
4. Compute $d = H(e||V)$;
5. Compute $\sigma = x \cdot d + k \bmod q$;
6. (e,σ) is the digital signature.

The ENC denotes a symmetric-key encryption algorithm. Similarly, we later denote ENC^{-1} as a decryption operation, which usually uses the same symmetric-key encryption algorithm. The signer sends $\langle V, e, \sigma \rangle$ to the receiver. To verify the message $M = C||V$ and the signature, the receiver needs to do following steps.

1. Compute $d = H(e||V)$;
2. Compute $R = \sigma P - dQ$;
3. Compute $C = ENC^{-1}(X(R), e)$;
4. Check the redundant information in C.

4.3 Threshold signature generation

Our event report signature generation scheme combines the ECPVS digital signature and Shamir secret sharing scheme [29] to generate the threshold signature. Examining the ECPVS protocol presented in Sect. 4.2, the signer has to have secret k and x. Considering a group of local sensors are the signer, the challenge of signature generation is how the group jointly constructs k and x (step 1 of ECPVS signature generation), the encryption of the content C (in step 3), and the calculation of *sigma* (in step 5). Note that any member of the group should not learn and reveal any information about k and x, assuming the adversary may capture all the communications inside the group.

We adopt Shamir's secret sharing scheme [29] to share system secret x. To achieve that, CA maintains a secret

polynomial: $f_x(y) = x + a_1 y + \cdots + a_{t-1} y^{t-1}$. Before being deployed, each sensor s_i receives a secret share of $f_x(y)$, which is denoted as x_i, and $x_i = f_x(s_i)$. Any t sensors can reconstruct the system private key: $x = \sum_{i=1}^{t} x_i l_i$, where l_i is the Lagrange coefficient. Any $t-1$ or less sensors, on the other hand, can not compromise system secret x because of the threshold property.

Shamir's secret sharing system discussed above, however, can not be used to share the secret random number k. The reason is that ECPVS signature scheme requires the signer should pick a different random k for a different signature. Otherwise, an adversary may easily derive system secret x by only capturing two signatures generated from the same k. To share a different random secret k among the group of sensors each time, we adopt the Joint Shamir Random secret sharing scheme [29]. This scheme allows all participating sensors to generate their own random secret polynomials (similar to $f_x(y)$) each time. To share a random secret k, each sensor in turn acts as a dealer to distribute the share of the secret (of his own polynomial) to the other members in the group. It should be emphasized that the polynomial shares must be distributed through the secure communication channels. We assume sensors already establish the pair-wise keys with their neighboring sensors by using existing schemes [6–10, 30]. In particular, sensor s_i generates its secret random polynomial $f_{s_i}(y)$, and distributes the share of secret $f_{s_i}(s_j)$ to sensor s_j ($1 \le j \le t$, $j \ne i$). Then, each sensor receives $t-1$ secret shares from the other $t-1$ sensor in the group, and one share of its own. By combining these t secret shares, each sensor s_i computes its own share of k, denoted as k_i, and $k_i = \sum_{j=1}^{t} f_{s_j}(s_i)$. The shared secret, as the random number k, is actually embedded in the polynomial that is the summation of t secret random polynomial generated by each of t sensors, $g(y) = \sum_{i=1}^{t} f_{s_i}(y)$. The secret k is determined by: $k = g(0)$. In this way, no sensor in the group knows the value of k. Any t sensors, however, can jointly reconstruct k by using Larrange interpolation: $k = \sum_{i=1}^{t} k_i l_i$. Again, l_i is the Lagrange coefficient.

With both k and x shared, the event report threshold signature generation scheme is illustrated in Fig. 1. We assume t sensors, $s_1, \ldots, s_t$, detect the event, denoted as $M = C||V$, where C can be the secret event measures and V can be general event description. We also assume s_1 is elected as the group leader. First, t sensors construct kP. Each sensor s_i sends its share $k_i l_i P$ to group leader s_1 (l_i is the Lagrange coefficient), which in-turn sums the t shares to get kP (by Lagrange interpolation), denoted as R. Then, s_1 broadcasts R to the rest $t-1$ sensors. Each sensor uses R to generate e and d as shown in Fig. 1, computes its share of the system signature: $\sigma_i = x_i l_i d + k_i l_i$, and send it to s_1 through the secure communication channel. The summation of t shares of signatures produces the system signature:

for (each sensor $s_i, i = 1, 2 \cdots, t$)
$s_i \rightarrow s_1 : P_i = k_i l_i P$

$s_1 \rightarrow s_2, s_3, \cdots, s_t : R = \sum_{j=1}^{t} P_j$ (i.e., $R = \sum_{j=1}^{t} k_j l_j P$)

for (each sensor $s_i, i = 1, 2 \cdots, t$)
$s_i : e = MAC(X(R), C)$
$s_i : d = H(e||V)$
$s_i : \sigma_i = x_i l_i d + k_i l_i$
$s_i \rightarrow s_1 : \sigma_i$

$s_1 : \sigma = \sum_{i=1}^{t} \sigma_i$

(i.e., $\sigma = \sum_{i=1}^{t} x_i l_i d + \sum_{i=1}^{t} k_i l_i = xd + k$)

Fig. 1 Event report threshold signature generation scheme by t sensor nodes, s_1,s_2,...,s_t

$\sigma = \sum_{i=1}^{t} x_i l_i d + \sum_{i=1}^{t} k_i l_i = xd + k$. Finally, s_1 sends (σ, e, V) to the destination, either the sink or other remote sensor node.

An important security measure of the above joint signature generation protocol is not to reveal any of system secrets, x, k, and individual sensor secret shares, s_i, x_i, at any step of the protocol. Otherwise, if the group leader is compromised, then system secrets are compromised. When kP is constructed to encrypt C (in step 2 of ECPVS), each sensor s_i submits $k_i l_i P$ instead of explicit k_i, so that secret share k_i is protected by the security property of discrete logarithm problem, i.e., it is infeasible to derive k_i from $k_i P$. When the signature σ is built (in step 5 of ECPVS), the partial signature σ_i submitted by each sensor is the linear combination of two unknown secrets x_i, k_i, so the group leader has no way to derive the values of x_i, k_i from σ_i. Overall, all secrets are well protected during the signature generation by the group.

4.4 Cost analysis

The t endorsing sensors have to jointly generate a random value k for each event report. To share a random k, each participating sensor s_i first generates its own random polynomial $f_{s_i}(y)$, and calculate the secret shares for other members in the group. For the group with t members, each sensor has to compute t shares of the $t-1$ degree polynomial, including the one for itself. We will show in the evaluation that the polynomial calculation is efficient for the motes. For the message complexity, each sensor sends $t-1$ secret shares to the $t-1$ members, and receives $t-1$ shares from the $t-1$ members. Therefore, each sensor has to send and receive $2(t-1)$ messages.

Note the share of k can be pre-computed. The group of sensors can run the secret sharing protocol at the idle time

before the event is detected, so the shares of a new k is ready for the next endorsement as long as the different events do not occur at the same location at the same time. Another way to reduce the communication overhead for k sharing is to eliminate the k sharing procedure by using pre-computation. If sensor nodes have enough storage space, CA can pre-compute different polynomials and pre-load the shares into the sensors during the deployment. Each share is associated with an index number. To endorse a new signature, the group of sensors only need to negotiate a new index, and use that share to construct a new random k. In this way, the message complexity can be reduced to the minimum.

After the shares of k are ready, the most expensive computation for each sensor s_i is one ECC point multiplication to compute P_i as shown in Fig. 1. For the message complexity, each sensor needs to send or receive two points and one scalar value, which includes its share P_i, the value of kP, and its share of σ.

The event report message consists of σ, e and V. Since V has the half size of e, the total message length is the size of two and half scalars. The computational cost to verify the report, as shown in Sect. 4.2, is two ECC point multiplications.

4.5 Security analysis

Our security analysis of the threshold signature generation scheme focuses on following two threats. We first check whether or not the adversary can infer the system secret by compromising one or more sensors and collaborating with other sensors in signature generation. Second, we examine the security resilience of secret k sharing because the compromise of k will lead to the whole system secret compromise. Note, the security resilience of sharing secret secret x by any t sensors is guaranteed by Shamir's secret sharing scheme. As long as there are no more than t sensors are compromised, there is no feasible solution to get x.

A compromised group leader certainly may cause greater security threat than other sensors since the leader collects more information, so the following analysis is based on the assumption that the group leader is compromised. The group leader (s_1) receives t shares of kP from other endorsing sensors and derive the value of kP, but the values of these points do not reveal any information of k_i or k due to the security property of ECC. The group leader s_1 also receives t shares of system signature. In each share, $\sigma_i = x_i l_i d + k_i l_i$, there are two unknown values: x_i and k_i. Any single or multiple shares combined does not reveal any information of x_i and k_i. Therefore, s_1 has no way to determine the system secret x and the random k without physically compromising the rest $t - 1$ endorsing sensors. As we can see, even though s_1 can be compromised, the adversary still cannot obtain the system secret to generate the signature for his injected data.

The shared random number k has a critical security role in PDF scheme. As we discussed previously, the compromise of k directly leads the compromise of system secret x. Therefore, any one or more (less than t) sensors must not get any information of k during the signature generate, otherwise the compromise attack may allow the adversary to acquire k. In Joint Shamir Random Secret sharing scheme [29], secret k is embedded in the polynomial that is the combination of each secret polynomial of t sensors. Each group member s_i holds a secret share of k, k_i. As long as at least one sensor is not compromised, the adversary can not get t secret shares, and thus can not reconstruct k. Further, since sensors do not directly send their secret shares k_i to the group leader, (instead they bind their k_i with x_i and the endorsed messages abstract d), the traffic monitoring does not give the adversary any chance to obtain the secret shares. Note the communication channel between sensors are encrypted to raise the security threshold and defend against other security attacks, including message injection and impersonate attacks.

One may wonder whether the compromised group leader can generate the signature for the forged messages because it can collect t partial signatures. This forged signature generation attempt, again, will fail because the partial signature submitted from each group member is bind with the endorsed event. In particular, the message abstract d is bind with each partial signature. If these partial signatures are used on a forged message, ECPVS verification will fail and the forged message will be immediately dropped by the forwarding sensors.

Finally, PDF scheme does not prevent the adversary's disruption attacks. The disruption attack happens when the adversary (by compromising one or more sensors) intentionally submits a corrupted partial secret or signature and then disturbs the signature generation. As the result, the generated threshold signature is invalid, and the legitimate event report will be dropped by the forwarding sensors. Several schemes have been proposed to identify the compromised sensors in prior studies [31–33], the security solution for defending against the disruption or false negative attacks is still an open research problem [2], and is considered as one of our future work. It should be emphasized that the disruption attack may trigger the system attention from the base station or sink because the network abnormality can be detected if many legitimate reports are dropped due to the attack. Then, the administration personnel can physically locate the compromised sensors and remove them from the network.

4.6 Probabilistic false data filtering

Given the event report with system signature, any intermediate forwarding sensor can easily verify the signature and decide whether or not to drop the packet. Theoretically, starting from the source node (s_1) to the destination, only one verification is enough to filter the possible false data packet. The signature verification at every hop is not necessary. However, considering the adversary's DoS attack can occur at any location in the network, one signature verification is not adequate because the adversary can inject the false data after the node that verifies the signature. Therefore, we propose the probabilistic false data filtering to balance the trade-off between computation overhead and the DoS attack prevention.

We denote p_f, a system wide parameter, as the en-route verification probability. Any intermediate forwarding sensor, with the probability of p_f, verifies the system signature by using the verification method presented in Sect. 4.2 . The verifying sensor first calculates $d = h(e||V)$, then deduces $R = \sigma P - dQ$ (P is the base point, and Q is the system public key). The value of X-coordinate of R is used to recover C, which is the part of original message M. Finally, the verifying sensor compares the redundant information in C with V. The event report message will be regarded authentic if the verification is successful. Otherwise, the message will be immediately dropped.

5 Performance evaluation

We evaluate our proposed PDF scheme by implementing all components on the real world experiment test-bed, including sensor confidential generation and pre-loading, security communication channel establishing, random secret number sharing, and threshold signature generation.

5.1 Experiment testbed and parameter setting

Our experiments use MICAz [34] motes as the sensor platform. MICAz is powered by an ATmega128 microcontroller, which features an 8MHz, 8-bit RISC CPU, 128K bytes flash memory (ROM) and 4K RAM. The RF transceiver on MICAz is IEEE 802.15.4/ZigBee compliant, and can achieve maximum 250kbps data rate. Our MICAz motes run TinyOS [35] version 1.1.15.

We implement ECC public key primitives on MICAz motes. We choose SECG recommended 160-bit elliptic curve, secp160r1, in our ECC implementation. The 160-bit ECC offers the same security level as the 1024-bit RSA does [36], which is a more popular public key scheme and widely used in e-commerce. The performance of threshold signature generation and public key verification directly determines the performance of PDF. The current ECC implementation in the public domain suffers very poor performance if ported directly. It is reported in [37] that it takes more than 30 s to generate a public key. To significantly reduce the computation time for ECC exponentiation, we have adopted a number of optimization techniques customized for the 8-bit architecture, including Hybrid Multiplication and Pseudo Mersenne modular reduction for large integer multiplication, Mixed Coordination for efficient ECC additions and doubling, etc. Due to the space limit, this paper omits the detail description of the optimizations. We refer interested readers to [38] for details. We summarize the key performance results in Table 1.

Table 1 The performance 160-bit ECC on MICAz mote, including fix point multiplication (FPM), random point multiplication (RPM), signature generation (Sign) and signature verification (Verify)

Platform	FPM	RPM	Sign	Verify
MICAz	1.24 s	1.35 s	1.35 s	1.96 s

We run the experiment in an office room with the dimension of 15 ft by 10 ft. The sensors are evenly placed on a table with the average distance of 2ft with each other. To achieve the better communication efficiency, we change the default TinyOS data packet payload size to 68 bytes (including 4-byte control information) from the original 29 bytes. This allows us to transmit an ECC public key (40 bytes) in one data packet. One possible trade-off for payload size extension is that the packet may suffer transmission errors more easily than that of the default size. As the result, the communication efficiency can be affected when packet loss happens. Our experiment results, however, show the packet loss is rare after the payload size extension. It could be the reason that our sensor deployment condition is too ideal to show the difference after payload size extension. It is one of our future work to study the communication efficiency in an outdoor and realistic deployment condition.

With all security components implemented, the program has the code size (ROM) of 35,108 bytes and the data size (RAM) of 2,648 bytes. Given the capacity of 128KB programming (ROM) size and 4KB data (RAM) size, we only use less than 30% of the programming size. The rest space can be reserved for other applications or future expansion. One may be concerned that we have consumed about 65% of the data size so that other applications may have memory shortage. One feasible solution is to move the constant variables (for ECC parameters) from RAM to on-board permanent storage (EPROM or flash). Further, more optimized and careful programming can also ease the memory shortage.

Our evaluation focuses on the time consumption, including the communication delay and the computation

delay. We do not explicitly give the performance of power consumption, because the combination of message complexity and time consumption can always be approximately translated to the power consumption. In the experiment, we have also adopted the simple scheduling scheme so that the probability for the packet corruption due to the collision is very small. During the experiment, we repeat each test for 20 times, and record the average time consumption. We finally discuss the PDF scheme message overhead and its scalability when the network size grows.

5.2 Evaluation of local pair-wise key establishment

To build the secure communication channels, pair-wise keys have to be established among the sensors. In the implementation, we adopt the ECC-based pair-wise key establishing scheme proposed in our previous work [39]. To ease the explanation, we show the slightly tweaked key establishment protocol in Fig. 2. The adjustment is to optimize the key establishing time consumption.

We suppose the key establishment protocol is running between two neighboring sensors u and v. Each sensor is pre-loaded with its public key and privacy key pair as long as the system security credentials. For example, sensor u is pre-loaded with q_u, Q_u and C_u, which are the private key, the public key and the public key certificate, respectively. Q_u can be verified from C_u by following formula: $Q_u = e_u C_u + Q$, where $e_u = hash(ID||C_u)$, and Q is the system public key. In general, C_u is used to verify the public key Q_u. Steps (1) and (2) of the protocol showed in Fig. 2 present how v verifies u. Interested readers are referred to [39] for detail explanation of the sensor certificate generation and verification.

After verifying sensor u, sensor v immediately sends Y_v to u after generating Y_v, so that the calculation of R_v by v and $q_u Y_v$ by u, the two time consuming ECC exponentiation, can be performed simultaneously. In step (7) and (8), only u, who has the private keyq_u, can recover r_v from ξ_v. Finally, u and v agree on r_v as the secret key.

(1) $u \rightarrow v : Q_u||C_u$
(2) v : verifies Q_u, selects a random r_v
(3) $v : Y_v = r_v P$
(4) $v \rightarrow u : \xi_v||Y_v$
(5) $v : R_v = r_v Q_u, \xi_v = X(R_v) \oplus r_v$
(6) $u : \widetilde{r_v} = X(q_u Y_v)$
(7) $v \rightarrow u : \xi_v$
(8) $u : r_v = \widetilde{r_v} \oplus \xi_v$
(9) u, v : agree on r_v

Fig. 2 ECC-based pair-wise key establishment scheme between two neighboring sensors: u and v

Our experiment shows it takes 4.1 s to share a pair-wise key between two MICAz motes. The performance can be improved under the circumstance. For example, Y_v in step (3) can be pre-computed. In that case, the optimized scheme only takes 1.5 s to establish the secret between two MICAz sensors.

In the next experiment, we test the time consumption for multiple neighboring sensors to establish keys with other sensors in the same neighborhood area (achieving full connectivity). Since the ECC operation time is much longer than the message transmission delay, we schedule the communications among the sensors in such a way that allows all sensors to do ECC operation simultaneously. In particular, each sensor first broadcasts its certificate and public key. After all certificates and public keys are received, the group of sensors simultaneously verify other certificates and compute the challenges for all other parties. Then, each sensor in turn transmits the challenges. After all challenges are received, the group of sensors simultaneously decrypt the challenge, and finally establish the pair-wise keys. Figure 3 shows the time consumption of both pair-wise key establishing schemes for up to 16 sensors. Even though the number of edges increase quadratically when the number of sensors increases, we find the overall time consumption for key establishment grows linearly. This is because sensors compute ECC exponentiation in parallel. When there are 16 sensors, it takes around 35 s to achieve full key connectivity. Our optimized key establishing scheme again shows the superior efficiency, it only takes 14 s to establish pair-wise key among 16 sensors, almost one third of time consumption compared to non-optimized scheme.

5.3 Evaluation of threshold signature generation

In this subsection, we evaluate the false data filtering performance. We first present the performance of the two

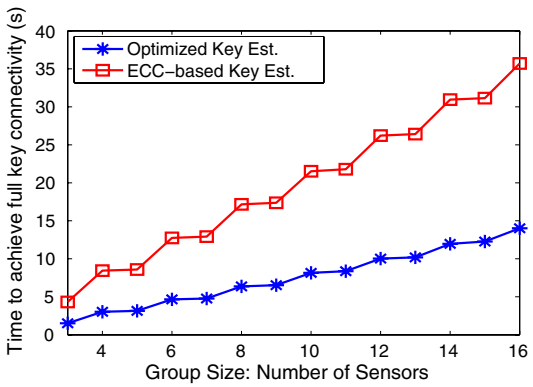

Fig. 3 The time duration for multiple neighboring sensors to achieve full key connectivity

components in PDF: threshold signature generation and signature verification. We then use the results to estimate the overall performance with different hop-by-hop authentication probabilities.

It is important that, in the threshold signature generation, the group of t local sensors need to share a different random secret k for each signature. Therefore, we first evaluate the cost for random secret (k) sharing. In the experiment, we first schedule all the motes to generate their random secret polynomials simultaneously, as well as the 20 byte secret shares for each of the other sensors in the group. Then, all the motes in turn unicast their secret shares to the corresponding sensors. We measure the time consumption in the whole process. The experiment results are illustrated in Fig. 4. We find the cost for sharing a random secret is not negligible but reasonable. For a group of 8 sensors, it takes only 1.8 s. The time consumption increases quadratically with the sensor group growing because the key graph edges increase with $O(n^2)$ (suppose n is the number of endorsing sensors). As the result, the communication complexity is $O(n^2)$. For a sensor group with 16 nodes, it then takes 5.8 s to share a random k.

Note the random k sharing protocol can be executed in the idle time before the event is detected, so that the random secret can be immediately used for endorsing the event upon detection. Therefore, the time duration for the threshold signature usually does not include the time delay for sharing k unless more than one different events occur simultaneously at the same location. Based on the above reason, our experiment for measuring the time delay for the threshold signature generation does not include the random k sharing time. We present the experiment results in Fig. 5. In general, the threshold signature generation is efficient because each endorsing sensor only needs to do one ECC point multiplication. With 8 local endorsing sensors, the time duration is 2.3 s. The time linearly increases to 3.3 s when the number of endorsing sensors becomes 16.

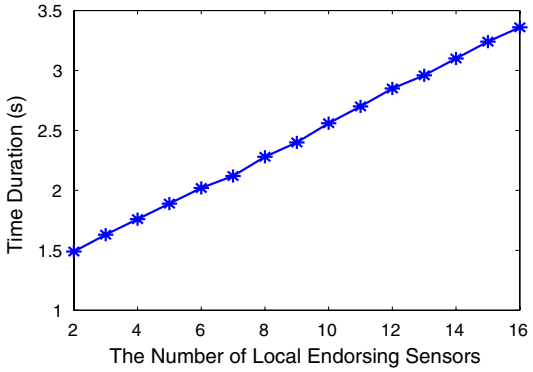

Fig. 5 The time duration for the group of local sensors to generate threshold system signature for the event report

The system signature verification is equivalent to an ECC signature verification operation. The verification time for an intermediate forwarding sensor is 1.96 s.

We are eager to investigate the overall performance of PDF, including threshold signature generation and the probabilistic false data filtering. In our evaluation, we assume that the event detecting sensors have already established pair-wise key with their neighbor endorsing sensors. We also assume these sensors have already shared a random secret k, which is used to generate the threshold signature. We fix the number of endorsing sensors to 16. Figure 6 demonstrates the overall performance of the false data filtering scheme under different hop-by-hop verification probabilities. As we can see, as long as the system parameter is properly selected, e.g., the verification probability is 10% or 20%, the overall performance of PDF is reasonably practical. Given the event report destination within less than 20 hops, the end-to-end delivery time is less than 10 s. While the delivery distance increases to 50 hops, the delivery time moderately increases to around 20 s.

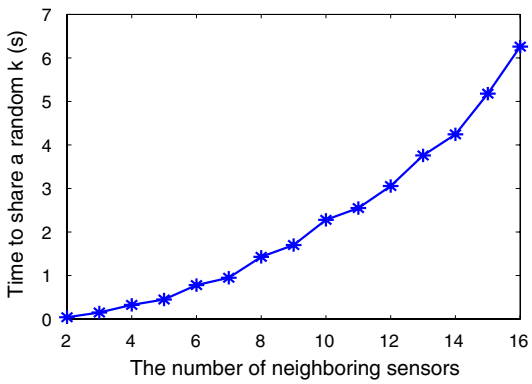

Fig. 4 The time duration for the group of sensors to share a random secret k

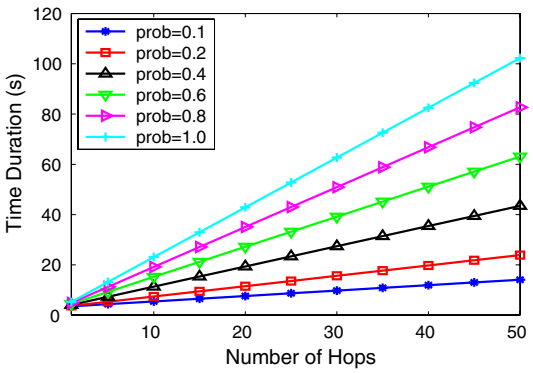

Fig. 6 The overall time duration of false data filtering performance under different probabilistic filtering value

5.4 PDF message overhead and its scalability

In addition to the time delay, PDF scheme also introduces extra messages to the sensor network. Since message complexity analysis for each group sensor was presented in the previous section, we here discuss the overall message overhead that PDF brings to the system. The overall message overhead is important because it shows how much communication cost the system has to pay to deploy PDF scheme to defend against the adversary's DoS attack.

The extra messages required in PDF scheme are used to share the random secret k, generate the group signature and the event signature attached to each event report message. Note we do not count the message overhead in pair-wise key establishing because it provides basic security infrastructure to the sensor network and can be included in any other security scheme besides PDF. Suppose there are t sensors in the group. As we indicated in the previous section, to share a random secret k, each sensor needs to send $t-1$ messages, so t sensors totally send $t(t-1)$ messages. As showed in Fig. 1, the signature generation scheme, each group sensor sends two messages to the group leader, and the group leader sends one message to the rest of group. The number of message combined is $3(t-1)$. In total, the number of extra messages to generate a signature in PDF scheme is $(t+3)(t-1)$.

As the ECPVS scheme shows in Sect. 4.2, once the signature is generated, the group leader sends the event report message in the format of (V, e, σ), where V is the public part of the message, e is the encrypted C, and σ is the group signature. Since the original message is $C||V$, the extra part sent in PDF scheme is just σ, which has the length of 20 bytes in 160-bit ECC system. Note this overhead is counted as per hop. If the average hop number is h, the total amount of message overhead for signature transmission is $20h$.

The above analysis reveals that the message overhead for signature generation is not related to the network size, and is only determined by the size of group (t). In event report transmission, PDF puts 20 bytes overhead in each event report. Considering that the average event report delivery distance may increase when the network size grows, PDF scheme may introduce the network size related message overhead. This overhead, however, can be very minimal as 20 bytes can be transmitted in the same message with the moderate payload size inflation.

6 Conclusion

Compared to the symmetric-key based solutions, the public-key based scheme offers much more solid security resilience. However, it is a challenge to design and implement the public-key scheme in resource constrained sensor networks. In this paper, we show our effort in designing the public-key based false data filtering scheme (PDF) in wireless sensor networks. Our scheme takes the advantage of recent progress in efficient implementation of ECC primitives on sensor devices. PDF allows each event report to be attached with a system signature jointly generated by a group of sensors nearby. This signature then is conveniently and efficiently verified by forwarding sensors along the routing path to the destination. We implemented all security components in PDF on MICAz sensor motes and run the protocol on our laboratory test bed. Our results demonstrated that the event report signature can be generated and delivered to the destination within 20 hops in 10 s. This result also supports we are the first to design and implement the practical public-key based false data filtering scheme in sensor networks.

Acknowledgments The authors would like to thank all the reviewers for their insightful comments and kind guidances to improve the paper. This project was supported in part by US National Science Foundation grants CNS-0721443, CNS-0831904, and CAREER Award CNS-0747108.

References

1. Yang, H., Ye, F., Yuan, Y., Lu, S., & Arbaugh, W. (2005, May). *Toward resilient security in wireless sensor networks*. Urbana-Champaign, IL: Mobihoc.
2. Ye, F., Luo, H., Lu, S., & Zhang, L. (2004). *Statistical en-route filtering of injected false data in sensor networks*. INFOCOM.
3. Yu, Z., & Guan, Y. (2006, April). *A dynamic en-route scheme for filtering false data in wireless sensor networks*. INFOCOM'06, Spain.
4. Zhu, S., Setia, S., Jajodia, S., & Ning, P. (2004, May). An interleaved hop-by-hop authentication scheme for filtering of injected false data in sensor networks. In *Proceedings of the IEEE symposium on security and privacy*, Oakland, CA.
5. Zhang, Y., Liu, W., Lou, W., & Fang, Y. (2006). Location-based compromise-tolerant security mechanisms for wireless sensor networks. *IEEE Journal on Selected Areas in Communications (Special Issue on Security in Wireless Ad Hoc Networks), 24*(2), 247–260.
6. Eschenauer, L., & Gligor, V. D. (2002, November). A key-management scheme for distributed sensor networks. In *Proceedings of the 9th ACM conference on computer and communication security*.
7. Chan, H., & Perrig, A. (2005, March). *Pike: Peer intermediaries for key establishment in sensor networks*. Miami, FL: INFOCOM.
8. Chan, H., Perrig, A., & Song, D. (2003, May). Random key predistribution schemes for sensor networks. In *IEEE symposium on Security and Privacy* (pp. 197–213). Berkeley, California.
9. Du, W., & Deng, J. (2003). *A pairwise key pre-distribution scheme for wireless sensor networks*. ACM CCS.
10. Liu, D., & Ning, P. (2003, October). *Establishing pairwise keys in distributed sensor networks*. Washington, DC: ACM CCS.
11. Liu, D. & Ning, P. (2005). Improving key pre-distribution with deployment knowledge in static sensor networks. *ACM Transaction on Sensor Networks, 20*, 1–32.
12. Traynor, P., Choi, H., Cao, G., Zhu, S., & T. L. Porta. (2006, April). *Establishing pair-wise keys in heterogeneous sensor networks*. Barcelona, Spain: INFOCOM.

13. Traynor, P., Kumar, R., Saad, H. B., Cao, G., & Porta, T. L. (2006, June). *Liger: Implementing efficient hybrid security mechanisms for heterogeneous sensor networks*. Uppsala, Sweden: Mobisys.
14. Zhang, Y., Liu, W., Lou, W., & Fang, Y. (2005, March). *Securing sensor networks with location-based keys*. New Orleans, Louisiana: WCNC'05.
15. Kong, J., Zerfos, P., Luo, H., Lu, S., & Zhang, L. (2001). Providing robust and ubiquitous security support for mobile ad hoc networks. In *Proceedings of the ninth international conference on network protocols* (p. 251). Washington, DC, USA: IEEE Computer Society.
16. Zhou, L., & Haas, Z. J. (1999). Securing ad hoc networks. *IEEE network, special issue on network security, 13*(2), 24–30
17. Amis, A. D., Prakash, R., Vuong, T. H. P., & Huynh, D. T. (2000). *Max-min D-cluster formation in wireless ad hoc networks*. INFOCOM.
18. Bandyopadhyay, S., & Coyle, E. (2003). *An energy-efficient hierarchical clustering algorithm for wireless sensor networks*. INFOCOM.
19. Bannerjee, S., & Khuller, S. (2001). *A clustering scheme for hierarchical control in multi-hop wireless networks*. INFOCOM.
20. Basagni, S. (1999). *Distributed clustering algorithm for ad-hoc networks*. I-SPAN.
21. Chatterjee, M., Das, S. K., & Turgut, D. (2002). *WCA: A weighted clustering algorithm for mobile ad hoc networks*. Cluster Computing.
22. Heinzelman, W. R., Chandrakasan, A., & Baladrishnan, H. (2002). An application-specific protocol architecture for wireless microsensor networks. *IEEE Transaction on Wireless Communication, 1*(4), 660–670.
23. Younis, O. & Fahmy, S. (2004). *Distributed clustering in ad-hoc sensor networks*. INFOCOM.
24. Intanagonwiwat, C., Govindan, R., & Estrin, D. (2000). *Directed diffusion: A scalable and robust communication paradigm for sensor networks*. MOBICOM.
25. Karp, B., & Kung, H. T. (2000). *GPSR: Greedy perimeter stateless routing for wireless neworks*. MOBICOM.
26. Ferreira, A. C., Vilaa, M. A., Oliveira, L. B., Wong, H. C., & Loureiro, A. A. (2005). *Networking-ICN* (pp. 449–458).
27. Newsome, J., Shi, E., Song, D., & Perrig, A. (2004). *The sybil attack in sensor networks: Analysis and defenses*. IPSN.
28. Certicom. (2004). Code and cipher. *Certicom's Bulletin of Security and Cryptography, 1*(3), 1–5.
29. Shamir, A. (1979). How to share a secret. *Communication of the ACM, 22*(11), 612–613.
30. Wang, H., Sheng, B., Tan, C. C., & Li, Q. (2008, June). Comparing symmetric-key and public-key based schemes in sensor networks: A case study for user access control. In *Proceedings of ICDCS*, Beijing, China.
31. Du, X. (2008). *Detection of compromised sensor nodes in heterogeneous sensor networks* (pp. 1446–1450). Beijing, China: ICC.
32. Zhang, Q., Yu, T., & Ning P. (2008). A framework for identifying compromised nodes in wireless sensor networks. *ACM Transactions on Information and System Security, 11*(3), 1–37.
33. Zhang, Y., Yang, Y., Jin, L., & Li, W. (2006). *Locating compromised sensor nodes through incremental hashing authentication*. San Francisco, CA: DCOSS.
34. Crossbow Technology INC. Wireless sensor networks. http://www.xbow.com/Products/Wireless_Sensor_Networks.htm.
35. Tiny OS. (2006). Tinyos 1.1.10. http://www.tinyos.net.
36. NIST. (2001, October). Key management guideline. In *Workshop document (DRAFT)*.
37. Malan, D. J., Welsh, M., & Smith, M. D. (2004, October). A public-key infrastructure for key distribution in tinyos based on elliptic curve cryptography. In *The first IEEE international conference on sensor and ad hoc communications and networks*, Santa Clara, CA.
38. Wang, H., & Li, Q. (2006, December). Efficient implementation of public key cryptosystems on mote sensors (Short Paper). In *International conference on information and communication security (ICICS)*. LNCS 4307 (pp. 519–528). Raleigh, NC.
39. Wang, H., & Li, Q. (2006). *Distributed user access control in sensor networks*. San Francisco, CA: DCOSS.

Author Biographies

Haodong Wang is currently a PhD candidate at Computer Science Department in the College of William and Mary. He got his BS from Tsinghua University and MS from Penn State University. His research interests are sensor network applications, security and privacy, security schemes on resource constrained devices, and wireless networks.

Qun Li is an assistant professor in the Department of Computer Science at College of William and Mary. He holds a PhD degree in Computer Science from Dartmouth College. His research interests include wireless networks, sensor networks, RFID, and pervasive computing systems. He received the NSF Career award in 2008.