PDF: A Public-key based False Data Filtering Scheme in Sensor Networks *

Haodong Wang and Qun Li
Computer Science Department
College of William and Mary
Williamsburg, VA 23187, USA
{wanghd,liqun}@cs.wm.edu

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.

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 resource for 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 much attention in the past several years. Most of prior work [17, 21, 16, 18], except [19], on sensor network report authentication and bogus data filtering mainly relies on symmetric key schemes. Ye *et al.* [17, 16] 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.* [21] 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 communication. 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 [19] proposed by Zhang *et al.* yields better security resilience. Unfortunately, the bilinear pairing

*This project is partially supported by the U.S. National Science Foundation under grant CCF-0514985.

based scheme is too expensive to be afforded by the low-power sensor hardware. As we will show in this paper, Elliptic Curve Cryptography (ECC) is more affordable than other public key schemes for sensor nodes. 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 this paper, we propose a Public-key based false Data Filtering scheme (PDF), which leverages threshold cryptography and ECC. 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 devise 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 [5]. 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 [3, 4, 7, 2, 8] 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 [4, 7] 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 [14, 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.* [20, 19] 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 [17, 21, 16, 18]. Zhu *et al.* [21] 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.* [17] 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.* [16]. 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 Enroute Filtering scheme [19] 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.

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.

The sensor network is managed by a Key Distribution Center (KDC), which is responsible for generating all security primitives and distributing the secret keys. During the sensor network deployment, KDC may preload the system secret information in each sensor node so that this information can be used in secret credentials establishment between the sensor nodes and as well as in future sensing tasks. We assume the sensor nodes have already established pairwise keys with their neighbor nodes, so that the communication between any two neighboring sensors are encrypted and protected.

An adversary is assumed to use all possible means to attack the sensor network. The potential attacks include communication eavesdropping, Man-In-The-Middle attacks, sensor compromise, and Denial-of-Service (DoS) attack. We assume the adversary is capable for monitoring all communications in the sensor network, which requires secure communication channel for all communication links. Due to the limited hardware resources, sensor nodes can be compromised upon capture. In this paper, we assume the adversary can retrieve all secret information from the compromised sensor. However, we assume that at most $t-1$ sensors can be compromised. The assumption is reasonable because compromising sensors takes time and effort. This paper focuses on the adversary DoS attack. The adversary may forge the event reports and inundate these report messages in the network to order to deplete the batter power of sensors and finally paralyze the network.

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 [1]. Third, we present the threshold endorsement false query 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 KDC maintains a system secret polynomial:

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

$a_0, a_1, ...a_{t-1}$ 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 [1] 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 (mod q), go back to the first step if the result is zero;
3. Compute $e = MAC(r, C)$;
4. Compute $d = H(e||V)$;
5. Compute $\sigma = x \cdot d + k$ mod q;
6. (e, σ) is the digital signature.

The MAC denotes the Message Authentication Code. The signer sends $< V, e, \sigma >$ 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 $C = MAC^{-1}(X(R), e)$;
4. Check the redundant information in C.

4.3 Threshold Signature Generation

Our event report threshold generation scheme combines the ECPVS digital signature and Shamir secret sharing scheme [11] to generate the threshold signature. Examining the ECPVS protocol presented in Section 4.2, the signer has to have secret k and x. Considering the group of local sensors are the signer, the key of signature generation is how the group of sensors jointly construct k and x, at the same time do not learn and reveal any information about k and x, assuming the adversary may capture all the communications inside the group.

We use Shamir's secret sharing scheme [11] to share system secret x. Similarly, KDC maintains a secret polynomial: $f_x(y) = x + a_1 y + \cdots + a_{t-1} y^{t-1}$. Before the deployment, 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 shares can reconstruct the system private key: $x = \sum_{i=1}^{t} x_i l_i$, where l_i is the Lagrange coefficient.

It is important in the ECPVS signature scheme that the signer has to pick a different random k for a different signature. Otherwise, an adversary only needs to capture two signatures generated from the same k and easily determines the private key x. To share a different random secret k among the group of sensors each time, we adopt the Joint Shamir Random secret sharing scheme [11]. This scheme requires 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. Since sensors already establish the pair-wise keys with their neighboring sensors, we can assume those secure communication channels are available. 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 $(1 \leq j \leq t, j \neq i)$. Then, each sensor receives $t-1$ secret shares from the other $t-1$ sensor in the group, and one share of its own. Finally, each sensor s_i sums the t shares and gets $k_i = \sum_{j=1}^{t} f_{s_j}(s_i)$. The shared secret, as the random number k, is $k = \sum_{i=1}^{t} f_{s_i}(0)$. In this way, no sensor in the group knows the value of k. However, the t sensors can jointly reconstruct k by using the similar formula: $k = \sum_{i=1}^{t} k_i l_i$. 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 endorsing sensors, $s_1, \cdots, s_t$, 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: $\sigma = \sum_{i=1}^{t} x_i l_i d + \sum_{i=1}^{t} k_i l_i = xd + k$. Finally, s_1 sends (σ, e, V) to the destination, either the sink or other remote sensor node.

4.4 Probabilistic False Data Filtering

Given the event report with system signature, any intermediate forwarding sensor can easily verify the sig-

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

Figure 1. Event report threshold signature generation scheme by t sensor nodes, $s_1, s_2, \cdots, s_t$.

nature 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 tradeoff between computation overhead and the DoS attack prevention.

We denote p_f, a system wide parameter, as the enroute verification probability. Any intermediate forwarding sensor, with the probability of p_f, verifies the system signature by using the verification method presented in section 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.

4.5 Cost and Security Analysis

We give the cost and security analysis as follows. 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 $t-1$ shares of the $t-1$ degree polynomial. 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 or 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.

It is also possible to eliminate the k sharing procedure for optimization under certain circumstances. If sensor nodes have enough storage space, KDC can precompute 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 Section 4.2, is two ECC point multiplications.

Our security analysis of the threshold signature generation scheme mainly focuses on whether or not the endorsing sensors can infer the system secret by collaborating with other sensors? 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 acquire the system secret to generate the signature for his injected data.

5 Performance Evaluation

We evaluate our proposed PDF scheme by implementing all components on the real world experiment testbed.

5.1 Experiment Testbed and Parameter Setting

Our experiments use MICAz [6] 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 [12] 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 [10], 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 [9] that it takes more than 30 sec 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 [15] for details. We summarize the key performance results in Table 1.

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

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).

To achieve a 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. Since we have already given the detail message complexity analysis for the scheme, 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.

5.2 Evaluation of Threshold Signature Generation

In this subsection, we evaluate the false data filtering performance. We first present the performance of the two 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 2. 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 seconds. 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 seconds 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

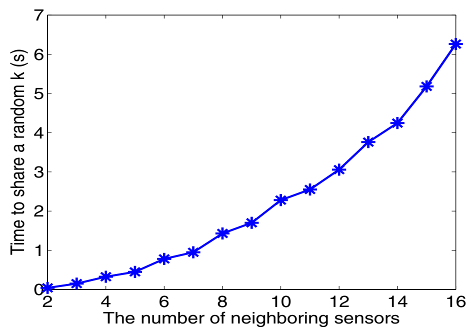

Figure 2. The time duration for the group of sensors to share a random secret k.

time delay for the threshold signature generation does not include the random k sharing time. We present the experiment results in Fig 3. 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 seconds. The time linearly increases to 3.3 seconds when the number of endorsing sensors becomes 16.

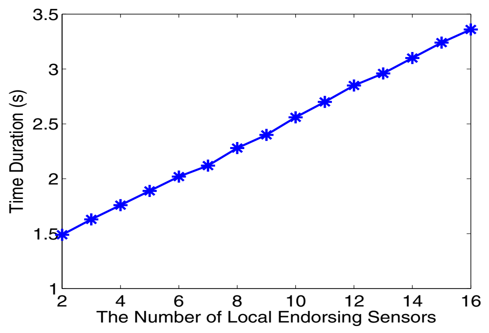

Figure 3. The time duration for the group of local sensors to generate threshold system signature for the user remote access query.

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

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. Fig. 4 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 10s. While the delivery distance increases to 50 hops, the delivery time moderately increases to around 20s.

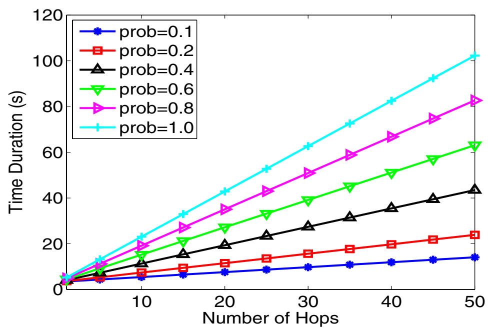

Figure 4. The overall time duration of false data filtering performance under different probabilistic filtering value.

6 Conclusion

In this paper, we show our effort in designing the public-key based false data filtering scheme (PDF) in wireless sensor networks. To achieve the goal, we first design the ECC-based local sensor pair-wise key establishment scheme. Based on that, we propose the threshold signature generation scheme that allows a number of event detecting sensor collaborate together and generate the system signature for the detected event, so that the intermediate sensors can efficiently verify. We implement the scheme on MICAz mote testbed. The security and performance analysis and the experimental results show that our PDF scheme is feasible for real application.

References

[1] Certicom. Code and cipher. *Certicom's Bulletin of Security and Cryptography*, 1(3):1–5, 2004.

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

[3] H. Chan, A. Perrig, and D. Song. Random Key Predistribution Schemes for Sensor Networks. In *IEEE Symposium on Security and Privacy*, pages 197–213, Berkeley, California, May 2003.

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

[5] L. Eschenauer and V.D. Gligor. A Key-management Scheme for Distributed Sensor Networks. In *Proceedings of the 9th ACM conference on Computer and Communication Security*, November 2002.

[6] Crossbow Technology INC. Wireless Sensor Networks. $http : // www.xbow.com /Products /Wireless\_Sensor\_Networks.htm$.

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

[8] D. Liu and P. Ning. Improving Key Predistribution with Deployment Knowledge in Static Sensor Networks. *ACM Transaction on Sensor Networks*, (20):1–32, December 2005.

[9] D.J. Malan, M. Welsh, and M.D. Smith. 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, October 2004.

[10] NIST. Key Management Guideline. In *Workshop Document (DRAFT)*, October 2001.

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

[12] TinyOS. Tinyos 1.1.10. *http://www.tinyos.net*, 2006.

[13] P. Traynor, H. Choi, G. Cao, S. Zhu, and T.L. Porta. Establishing Pair-wise Keys in Heterogeneous Sensor Networks. In *INFOCOM*, Barcelona, Spain, April 2006.

[14] P. Traynor, R. Kumar, H. B. Saad, G. Cao, and T. L. Porta. Liger: Implementing Efficient Hybrid Security Mechanisms for Heterogeneous Sensor Networks. In *Mobisys*, Uppsala, Sweden, June 2006.

[15] H. Wang and Q. Li. Efficient Implementation of Public Key Cryptosystems on Mote Sensors (Short Paper). In *International Conference on Information and Communication Security (ICICS), LNCS 4307*, pages 519–528, Raleigh, NC, Dec. 2006.

[16] H. Yang, F. Ye, Y. Yuan, S. Lu, and W. Arbaugh. Toward Resilient Security in Wireless Sensor Networks. In *Mobihoc*, Urbana-Champaign, IL, May 2005.

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

[18] Z. Yu and Y. Guan. A Dynamic En-route Scheme for Filtering False Data in Wireless Sensor Networks. In *INFOCOM*, Spain, April 2006.

[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 (Special Issue on Security in Wireless Ad Hoc Networks)*, 24(2):247–260, Feb 2006.

[20] Y. Zhang, W. Liu, W. Lou, and Y. Fang. Securing Sensor Networks with Location-based Keys. In *WCNC*, New Orleans, Louisiana, March 2005.

[21] S. Zhu, S. Setia, S. Jajodia, and P. Ning. An Interleaved Hop-by-hop Authentication Scheme for Filtering of Injected False Data in Sensor Networks. In *Proc. IEEE Symposium on Security and Privacy*, Oakland, CA, May 2004.