Commutative Cipher Based En-route Filtering in Wireless Sensor Networks

Hao Yang and Songwu Lu
Computer Science Department, UCLA
Los Angeles, CA 90095-1596, U.S.A.
E-mails: {hyang, slu}@cs.ucla.edu

***Abstract*— Wireless sensor networks offer unprecedented capabilities to monitor the physical world by reporting the occurrence of interested events. Unfortunately, such networks are vulnerable to event fabrication attacks, in which the compromised nodes inject bogus reports into the network, fabricating non-existent events "appearing" at arbitrary locations. Most existing security solutions to these attacks need to share symmetric keys among sensor nodes. In this paper, we propose a Commutative Cipher based En-route Filtering scheme (CCEF) that drops fabricated reports en-route without symmetric key sharing. In CCEF, the source node establishes a secret association with the base station on a per-session basis, while the intermediate forwarding nodes are equipped with a witness key. Through the usage of a commutative cipher, a forwarding node can use the witness key to verify the authenticity of the reports without knowing the original session key. As a result, CCEF can achieve stronger security protection than the existing symmetric key sharing approach.**

I. Introduction

Wireless sensor networks offer unprecedented capabilities to monitor the physical world and enable a variety of applications such as military surveillance, and vehicle safety monitoring. In a typical sensor network, the terrain under investigation is covered by a large number of sensors nodes with embedded sensing, computation and wireless communication capabilities. When an event of interest, such as enemy tanks or vehicles, occurs, nearby sensors detect the event and generate a data report on the event. The report is then forwarded hop-by-hop to the base station. In response, the network operators may send task forces or announce safety warnings to the location of the event. Security is an essential requirement for these mission-critical applications in an adverse environment.

Due to the sheer size of the network, individual sensor nodes are typically unattended, and thus prone to capture and security compromise. Consequently, the network is vulnerable to the security threat of *event fabrication attacks*, in which compromised nodes report *nonexistent* events "appearing" at arbitrary locations in the field, as illustrated in Figure 1. Left undetected, these fabricated reports will be delivered to the base station over potentially many hops. They not only produce false alarms that may cause the operators to dispatch response task-force to the "victim" locations, but also waste significant network resources, such as energy and bandwidth, in delivering them to the base station. The conventional node-based authentication mechanisms are not sufficient to defeat such attacks, because a compromised node already has enough keying materials to be authenticated.

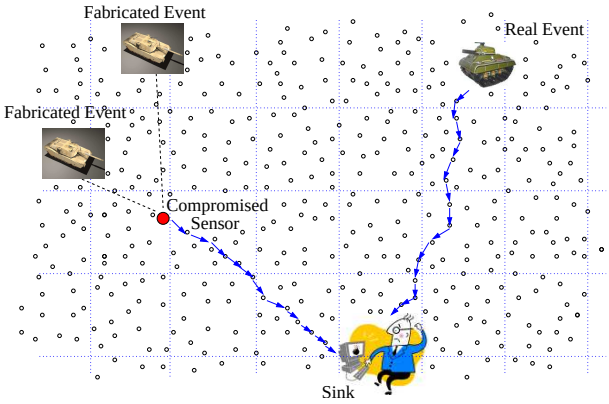

Fig. 1. Sensor networks are vulnerable to event fabrication attacks, in which the compromised nodes fabricate reports on non-existent events at arbitrary locations.

To minimize the grave damage, the fabricated reports should be dropped en-route as early as possible, and the few eluded ones should be further rejected at the base station. Several security solutions [12], [15] have recently been proposed for this purpose. In SEF [12], a report is forwarded if and only if it is endorsed by multiple nodes, using keys from different partitions in a global key pool. Hop-by-Hop Authentication [15] detects fabricated reports through interleaved authentication. While each of these designs has its own merits, they can address only a fixed threshold of compromised nodes. The fundamental reason is that these designs follow the *symmetric key sharing* approach in achieving the en-route filtering capability. In this approach, a credential (e.g., MAC) that endorses a report is generated and verified using the same symmetric key. To enable en-route filtering, the forwarding nodes must share keys with the source nodes. As a result, there exists an inherent limitation of resiliency to multiple compromised nodes. Because the keys are symmetric, the compromised nodes may abuse their verification keys to generate credentials and endorse a bogus report. By compromising more and more nodes, the attacker may eventually inject fabricated reports without being detected.

In this paper, we propose a novel Commutative Cipher based En-route Filtering scheme (CCEF) that defends against event fabrication attacks without symmetric key sharing among sensor nodes. CCEF exploits the typical operational mode of

query-response in sensor networks, and installs security states in the nodes in an on-demand manner. Specifically, in CCEF, each node has a unique ID and is preloaded with a unique node key. The base station initiates a query-response session by sending out a query to task specific sensor nodes to report their sensing results. The base station prepares two keys for each session: one *session key* and one *witness key*. The session key is securely sent to source node, i.e., the node tasked to generate reports, while the witness key is in plaintext and recorded by all intermediate nodes. A legitimate report is endorsed by a node MAC jointly generated by the detecting nodes using their node keys, and a session MAC generated by the source node using the session key. Through the usage of a commutative cipher, a forwarding node can use the witness key to verify the session MAC, without knowing the session key, and drop the fabricated reports. The base station further verifies the node MAC in the report that it receives, and refreshes the session key upon detection of compromised nodes. .

Our security analysis shows that as long as the adversary has not directly compromised t nodes at one location, he cannot fabricate any events "appearing" at this location without being detected. In contrast, this can be hardly achieved in the symmetric key sharing approach.

II. Background

In this section, we describe the models and assumptions in our design, and provide background knowledge on commutative ciphers.

A. Models and Assumptions

We consider a large-scale sensor network in which the nodes do not move after initial deployment. Each sensor node has only limited sensing range, and the network is dense to provide both fine-grained sensing information and desirable robustness against node failures. The sensor nodes are battery-powered. In order to prolong the network lifetime, it is crucial to conserve the energy consumptions at these nodes. The network user queries the network through a base station, which is a powerful workstation with much stronger computation and storage capabilities than the sensor nodes.

We assume that certain localization components [13] exist in the system, so that each node can obtain its location after deployment. Such location-awareness is required by most monitoring applications to determine the locations of events. We also assume that the wireless link is bi-directional in that if node A is in the transmission range of node B, node B is also in the transmission range of node A.

We focus on the security threat of event fabrication attacks in this work. The attacker may compromise multiple sensor nodes in the network, and we do not impose any bound on the number of compromised nodes. Once a node is compromised, the attacker can extract all secret keys, data, and codes stored on the node, and have full control over its actions. The attacker is allowed to combine the keys obtained from multiple nodes in launching the attacks. However, we assume that the attacker cannot compromise the base station, and the system bootstrapping phase is secure. To protect the network from naive impersonation attacks, we assume that some basic security mechanisms of neighbor authentication [7], [14] already exist.

B. Commutative Cipher

Commutative cipher has been widely used in the literature for various purposes, to name a few, privacy protection [3], [5], secure e-commerce [4], and oblivious information sharing [2]. A cipher *CE* is commutative if and only if it satisfies the following property: for any message M and any two keys K_1 and K_2,

$$CE(CE(M, K_1), K_2) = CE(CE(M, K_2), K_1) \quad (1)$$

That is, when one uses the commutative cipher to apply two encryption operations on a message, the encryption sequence does not change the ultimate result. In general, this property holds for any finite number of encryption operations.

A simple example of such commutative ciphers is the XOR stream cipher. However, this stream cipher cannot be used to encrypt more than one message, without revealing the key. In fact, most known symmetric key ciphers are not commutative. The two popular commutative ciphers use the Pohlig-Hellman algorithm based on Elliptic Curve Cryptography (ECC), and the Shamir-Omura algorithm based on RSA cryptography, respectively. Due to space constraints, we do not elaborate on the cryptography implementations of commutative ciphers in this paper. Interested readers are referred to [4], [6], [11] for more details on this topic.

III. Commutative Cipher Based En-route Filtering

In this section, we present the design of a novel commutative cipher based en-route filtering scheme (CCEF). CCEF exploits a bootstrapping phase to establish trust between individual sensor nodes and the base station. In the operational phase, the base station can initiate a query-response session and install per-session security states in the sensor nodes at any time. The tasked sensor nodes response by generating and endorsing data reports on their sensing results. When the reports are forwarded to the base station, each intermediate node verifies the authenticity of the reports, and filters the fabricated ones. The base station further verifies the reports that it receives, and reacts to the compromised nodes by refreshing the session state.

A. Bootstrapping

In CCEF, each node has a unique ID, and is preloaded with a unique key that is shared only with the base station, henceforth called *node key*. After the nodes are deployed over the terrain (e.g., via aerial scattering), the system bootstraps with a localization component, from which each node can obtain its own location. The nodes then report their locations to the base station, which ends the bootstrapping phase.

The node locations will be used by the base station to verify the authenticity of the reports, as we will describe later. Note that a sensor node does not store the locations of any other

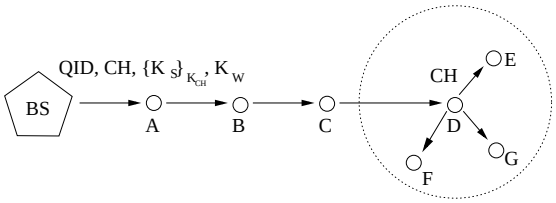

Fig. 2. The base station initiates a session with a query. The query includes an encrypted session key and a witness key. The cluster header (node D in this example) decrypts the session key, and the intermediate nodes record the witness key.

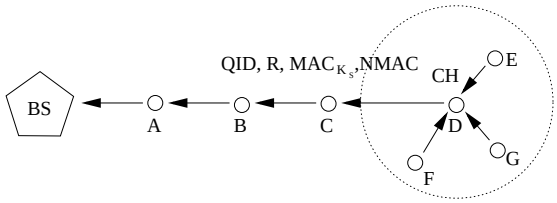

Fig. 3. The report carries a session MAC and a node MAC, and is forwarded to the base station in the reversed path. Each forwarding node probabilistically uses the witness key to verify the session MAC. The base station further verifi es the node MAC.

nodes. To decrease the communication overhead, the location reports can be aggregated or piggybacked in other messages.

B. Session Setup

In the operational phase, the sensor nodes are tasked by the base station through queries. A typical query in sensor network looks like "How many tanks are there in the geographic region X". For this purpose, each query specifies the interest of the network user, expressed in multiple attribute-value pairs such as *type=tank* and *rect=[100,100,200,200]*. We consider queries with known interest of locations, such as the example given above, for now to simplify the presentation. In Section III-G, we will extend the design to handle general queries that do not have specific location interest, e.g., "Is there any tank in the field".

Because the base station knows the location of all nodes, it can randomly select one sensor node at the location of interest as a local cluster header (CH). As shown later, the cluster header will lead the collaborative sensing in the local neighborhood, and aggregate the sensing results from individual sensor nodes into a single report. Such local aggregation is adopted in most sensor network applications to avoid duplicate reports and save energy resources,

For each session, the base station prepares two keys K_s and K_w such that

$$CE(M, K_w) = CE^{-1}(M, K_s) \tag{2}$$

in which CE and CE^{-1} are the encryption and decryption algorithm of a commutative cipher, respectively. Note that the reversed property, $CE(M, K_s) = CE^{-1}(M, K_w)$, does not hold because CE and CE^{-1} are not commutative with respect to each other. The key K_s is called the *session key*, and key K_w is called the *witness key*. These two correlated keys uniquely define the security states associated with a session.

The query-response session is initiated by the base station as follows. The base station constructs a query that includes not only the application-specific interests, but also four additional fields: 1) a unique *QID* (Query ID); 2) the CH node ID; 3) the session key K_s encrypted by CH's node key, i.e., $\{K_s\}_{K_{CH}}$; 3) the witness key K_w as plaintext. The query message is authenticated by an authentication scheme such as μTESLA [10].

As shown in Figure 2, the base station sends out the query message, which is forwarded hop-by-hop to CH, e.g., through geographic routing protocols [8]. Each intermediate node stores the tuple (QID, K_w) for future verification purpose. When CH (e.g., node D) receives the query, it decrypts the session key K_s, and then forwards the query to its local neighbors (e.g., nodes E, F, and G).

C. Report Generation

The tasked sensor nodes response to the query by collaborative sensing and generation of data reports. We ignore the details of collaborative sensing here, and start with a scenario in which the sensor nodes in the local neighborhood have reached an agreement on the event description R, i.e., the content of the report. The message exchange between local neighbors is authenticated.

In CCEF, each legitimate report is endorsed by two MACs. The first one is called *session MAC*, generated using the session key, while the second one is called *node MAC*, generated using the node keys. These two MACs are generated as follows. Each sensor node that agrees on the event uses its node key to generate a MAC on R, and sends the MAC to CH. CH then randomly select t MACs that it has received, and compresses them into a node MAC using standard XOR scheme. In the example shown in Figure 3, node D generates the compressed node MAC as:

$$NMAC = MAC(R, K_E) \oplus MAC(R, K_F) \oplus MAC(R, K_G).$$

In addition, CH uses the session key K_s, which it decrypted from the query, to endorse the report with a session MAC:

$$MAC_{K_s} = CE(R, K_s). \tag{3}$$

After CH has computed the above two MACs, it constructs a final report that includes the content, the session MAC, and the node MAC. The IDs of the t endorsing nodes are also included in the report, so that the base station can verify the node MAC. Finally CH disseminates the report towards the base station

D. En-route Filtering

The report is forwarded along the reversed path as the query traverses. When an intermediate node receives a report, it first extracts the QID field and checks whether it has stored the tuple (QID, K_w). If not, it drops the report. Otherwise, it uses

the corresponding witness key K_w to verify the session MAC. Based on Equations 1, 2, and 3, we know that:

$$\begin{aligned} CE(MAC_{K_s}, K_w) &= CE(CE(R, K_s), K_w) \\ &= CE(CE(R, K_w), K_s) \\ &= CE(CE^{-1}(R, K_s), K_s) \\ &= R \end{aligned} \quad (4)$$

This way, a node can use the witness key to verify a session MAC, without knowing the session key. This is exactly where the name of "witness key" comes from. The forwarding node drops the report if the session MAC does not match its witness key as in Equation 4. It also keeps a local cache of previously seen reports to prevent replay attacks.

Note that if every forwarding node always verifies all reports passing through it, the fabricated reports with incorrect session MAC can be dropped one hop after it is injected. However, a legitimate report will be verified at each hop. Because the commutative cipher computation is quite heavy, such excessive verification leads to energy inefficiency. Therefore, we adopt a probabilistic approach in which a forwarding node verifies a report with a probability of

$$P = \frac{1}{\alpha h} \quad (5)$$

where α is a system parameter, and h is the number of hops from CH to the base station. In practice, h can be estimated by the base station, or provided by the underlying routing protocol. In Section IV, we will analyze the energy efficiency of such probabilistic filtering.

E. Base Station Verification and Reaction

The base station knows the session key and all node keys. When it receives a report, it first verifies the session MAC. If the session MAC is incorrect, it rejects the reports without further actions. Otherwise, the base station knows that the report must come from CH, because CH is the only node that can decrypt the session key from the query message. The base station then verifies whether the node MAC is correct, and whether the t endorsing nodes are indeed in the local neighborhood of CH. If either check fails, the base station knows that either CH or one endorsing node is compromised. This is because each sensor node has only limited sensing and communication range, so it cannot communicate with other faraway nodes or detect remote events. In such cases, the base station drops the report and reacts to the node compromise as follows. It first sends out a session revocation message to inform the intermediate nodes to invalidate the session state, i.e., tuple (QID, K_w). It then sends out a node revocation message to inform the nodes at the victim location to quarantine CH as well as the t endorsing nodes. When the node density at that location decreases below a threshold, new nodes may be deployed to maintain the monitoring capability.

F. Session Maintenance and Teardown

The sensor nodes are prone to failures. Therefore, the base station periodically probes the current CH to maintain the session. When the current CH fails, the base station selects a new node as CH, and securely sends the session key to it as before. Each session is associated with a duration, which specifies how long the query is valid. When the session expires, intermediate nodes remove the session states.

G. Querying Unknown Locations

Now we briefly describe how to extend the above design to address general queries that do not have *a priori* information on the locations of interest. An example of these queries is "Is there any tank in the field". In such cases, the base station floods a probe message in the network that specifies the application interests. The probing phase lasts for only a short period of time. The sensor nodes whose sensing results match the interest send out an acknowledgment message to the base station. The acknowledgment message is forwarded unconditionally; however, to be accepted by the base station, it must be jointly endorsed by t nodes using their node keys. After the base station accepts an acknowledgment message, it knows the location of the reporting nodes, and thus can initiate a session as discussed above.

IV. Analysis

In this section, we evaluate the security strength and energy saving performance of CCEF through analysis.

A. Security Analysis

The security protection offered by CCEF is characterized by the following proposition.

Proposition 1: As long as the adversary has not directly compromised t nodes at one location, he cannot fabricate any events "appearing" at this location without being detected. Furthermore, in such cases, the fabricated reports are dropped quickly along the forwarding paths.

We briefly describe why this proposition is true. Because the session is always initiated by the base station, any unsolicited event reports will be dropped. Without loss of generality, we consider one particular session. There are two possible cases: CH is uncompromised, or CH is compromised. 1) In the uncompromised CH cases, none of the compromised nodes knows the session key. Thus the session MAC in the fabricated reports must be incorrect. With high probability, one of the forwarding nodes will detect it and drop the report. In the worst case, the base station can still detect it. 2) In the compromised CH cases, the adversary can correctly generate the session MAC, thus the fabricated reports will not be dropped en-route. However, as long as the adversary has not compromised t local nodes, he cannot forge the node MAC. Thus the fabricated reports are eventually detected by the base station, and the compromised CH is revoked.

B. Energy Saving Analysis

Now we analyze how much energy we can save by filtering fabricated reports using CCEF. We assume that all sensor nodes have the same hardware and use the same transmission power. We use e_1 to denote the energy consumed in transmitting a report (including the MACs), and e_2 to denote the energy

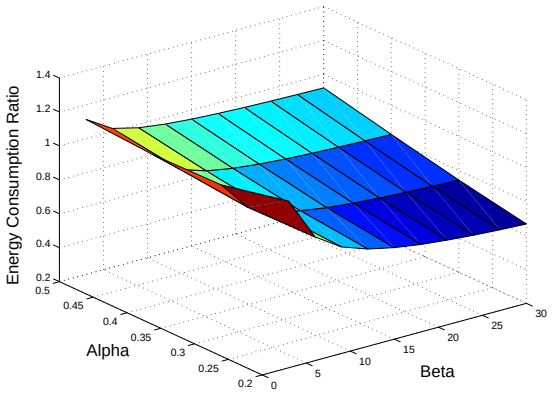

Fig. 4. CCEF can significantly save energy in large-scale networks by filtering the fabricated report early in deliver.

consumed in one commutative cipher computation. Let h be the number of hops in a forwarding path, and the normalized legitimate traffic and fabricated traffic be 1 and β, respectively. Without any protection, the energy consumption is:

$$E = (1 + \beta)he_1$$

Based on Equation 5, it is not hard to see that with CCEF, a fabricated report is dropped after αh hops on average and verified only once. A legitimate report is never dropped yet verified for $1/\alpha$ times. Thus, the energy consumption is:

$$E_C = (1 + \alpha\beta)he_1 + (\frac{1}{\alpha} + \beta)e_2$$

The energy savings that CCEF achieves can be best shown by the ratio of energy consumptions in these two cases.

$$\rho = \frac{E_c}{E} = \frac{(1 + \alpha\beta)he_1 + (\frac{1}{\alpha} + \beta)e_2}{(1 + \beta)he_1}$$

As we can see, when $h \to \infty$ and $\beta \to \infty$, we have an asymptotic performance of $\rho = \alpha$, where α is a design parameter (Equation 5). This clearly shows that CCEF can significantly save energy in a large-scale network by dropping large amount of fabricated reports en-route.

We also use empirical measurement results [1], [10], [9] on current-generation Mica2 sensor nodes to quantify the above results. Based on the real measurement, we can estimate that $e_1 = 1.15$ mJ (transmitting and receiving a 40-bytes report at 19.2 Kbps), and $e_2 = 90$ mJ (40-bit exponent, taking 3 seconds). We fix h as 100 and plot in Figure 4 the energy consumption ratio with different values of α and β. We can see that even with a relatively heavy commutative cipher, CCEF can save energy as high as 32% when the attacker injects large amount of fabricated reports into the network.

V. Conclusion

To this end, we have presented and evaluated the design of a commutative cipher based en-route filtering scheme for wireless sensor networks. CCEF differs from existing security solutions in that it decouples base station verification from en-route filtering, and does not share any symmetric keys between the sensor nodes. In CCEF, the monitoring capability is protected by the secrets shared between the sensor nodes and the base station, while the en-route filtering capability is achieved through a partial proof of the secret association. As a result, CCEF can provide much stronger security protection against compromised nodes than the symmetric key sharing based designs.

References

[1] Xbow sensor networks. http://www.xbow.com.

[2] R. Agrawal, A. Evfimievski, and R. Srikant. Information sharing across private databases. In *ACM SIGMOD*, 2003.

[3] F. Bao, R. Deng, and P. Feng. An efficient and practical scheme for privacy protection in the e-commerce of digital goods. In *ICISC*, 2000.

[4] F. Bao, R. Deng, P. Feng, Y. Guo, and H. Wu. Secure and private distribution of online video and some related cryptographic issues. In *ACISP*, 2001.

[5] S. chi Cheung, H. fung Leung, and C. Wang. A commutative encrypted protocol for the privacy protection of watermarks in digital contents. In *Hawaii International Conference on System Sciences*, 2004.

[6] W. Diffie and M. Hellman. New directions in cryptography. *IEEE Trans. on Information Theory*, 22(6):644–654, November 1976.

[7] L. Eschenauer and V. Gligor. A key-management scheme for distributed sensor networks. In *ACM CCS*, 2002.

[8] B. Karp and H. T. Kung. GPSR: Greedy perimeter stateless routing for wireless networks. In *ACM MOBICOM*, 2000.

[9] D. Malan. Crypto for tiny objects. Technical Report TR-04-04, Harvard University, 2004.

[10] A. Perrig, R. Szewczyk, V. Wen, D. Culler, and D. Tygar. SPINS: Security protocols for sensor networks. In *ACM MOIBCOM*, 2001.

[11] W. Stallings. *Cryptography and Network Security: Principles and Practice*. Prentice Hall, 1998.

[12] F. Ye, H. Luo, S. Lu, and L. Zhang. Statistical en-route filtering of injected false data in sensor networks. In *INFOCOM*, 2004.

[13] F. Zhao, J. Liu, Q. Huang, and Y. Zou. Fast and incremental node localization in ad hoc networks. Technical Report P-2003-10265, PARC, 2003.

[14] S. Zhu, S. Setia, and S. Jajodia. LEAP: Efficient security mechanism for large-scale distributed sensor networks. In *ACM CCS*, 2003.

[15] S. Zhu, S. Setia, S. Jajodia, and P. Ning. An interleaved hop-by-hop authentication scheme for filtering false data in sensor networks. In *IEEE SSP*, 2004.