Poster Abstract: A Dynamic En-route Scheme for Filtering False Data Injection in Wireless Sensor Networks

Zhen Yu and Yong Guan
Department of Electrical and Computer Engineering
Iowa State University, Ames, IA 50010
{yuzhen, yguan}@iastate.edu

Categories and Subject Descriptors: C.2.2 [Computer-Communication Networks]: Network Protocols, Wireless Communications
General Terms: Algorithm, Design, Security
Keywords: Wireless Sensor Networks, False Data Injection

1. INTRODUCTION

In sensor networks, adversaries can inject false data reports containing bogus sensor readings or nonexistent events from some compromised nodes. Such attacks may not only cause false alarms, but also drain out the limited energy of sensor nodes. Several existing false reports filtering schemes such as Statistical En-route Filtering (SEF) by Ye, Luo, et al., Interleaved Hop-by-hop Authentication Scheme by Zhu, Setia, et al., and Commutative Cipher based En-route Filtering (CCEF) by Yang and Lu, either cannot deal with dynamic topology of sensor networks or have limited filtering capacity.

We propose a dynamic en-route filtering scheme for false data injection attacks in wireless sensor networks. In our scheme, a legitimate report is endorsed by multiple sensing nodes using their distinct authentication keys from one-way hash chains. Cluster head uses *Hill Climbing* approach to disseminate the authentication keys of sensing nodes along multiple paths toward the base station. *Hill Climbing* guarantees that the nodes closer to a cluster hold more authentication keys for the cluster than those nodes farther from it do and the number of keys held by each forwarding node can be balanced. In filtering phase, each forwarding node validates the authenticity of the reports and drops those false reports. Compared to existing schemes, our scheme can better deal with dynamic topology of sensor networks. Analytical and simulation results show that our scheme can drop false reports earlier even with a lower memory requirement and tolerate more compromised nodes. Our scheme also outperforms others in term of energy efficiency, especially for large sensor networks.

2. SYSTEM MODEL AND GOALS

We consider such a scenario: After deployment, sensor nodes form a number of clusters, each of which contains n nodes. In each cluster, one node is elected as *Cluster Head* (CH). To balance energy consumption, all nodes of the same cluster may take turn to play the role of cluster head. An event should be detected by multiple sensing nodes simultaneously. When some event occurs, these sensing nodes send their sensing reports to the cluster head, which would aggregate these sensing reports, generate a final report, and send it to the *Base Station* (BS) on behalf of those sensing nodes. If nodes fail or change their state between active mode and sleeping mode, the topology of sensor network will be changed.

Sensor nodes may be compromised or physically captured. Adversaries can easily inject false reports into sensor networks through those sensor nodes compromised or captured. Considering the limited memory and computation capacity of sensor nodes, we set the following goals for the design of our scheme:

1. False reports can be detected and dropped as early as possible;
2. It can tolerate a larger number of compromised nodes;
3. It has low memory requirement and incurs low computation and communication overhead to sensor nodes;
4. It can deal with dynamic topology of sensor networks and is scalable for large-scale sensor networks.

3. OUR SCHEME

Our scheme consists of three phases: *Pre-deployment Phase*, *Post-deployment Phase* and *Filtering Phase*. Pre-deployment phase is executed only once, but the other two will be performed repeatedly.

Pre-deployment Phase: Before deployment, each node is preloaded with a seed authentication key and $l+1$ secret keys that include l y-key and one z-key. These two kinds of secret keys are used to encrypt the authentication key of the node and randomly picked from two global key pools. Meanwhile, each node generates a one-way hash chain from its seed authentication key. As shown in Figure 1, $y_1^{v_i}, \cdots, y_l^{v_i}$ and z^{v_i} are secret keys of sensing node v_i. And v_i generates a one-way hash chain such as $k_m^{v_i}, \cdots, k_1^{v_i}$ from its seed key $k_m^{v_i}$, where $k_j^{v_i} = h^{m-j}(k_m^{v_i})$ and $k_1^{v_i}$ is used first.

Post-deployment Phase: Before sending reports, each cluster head disseminates the authentication keys of all nodes in its cluster to the forwarding nodes. Each authentication key is encrypted by the corresponding node using its $l+1$ secret keys and these encrypted authentication keys are encapsulated in message $K(n)$. For example, in Figure 1 the cluster head CH may send a message $K(n)$ as follows:

SenSys'05, November 2–4, 2005, San Diego, California, USA.
ACM 1-59593-054-X/05/0011.

$$K(n) = \{ v_1, j_1, \{k_{j_1}^{v_1}\}_{y_1^{v_1}}, \cdots, \{k_{j_1}^{v_1}\}_{y_l^{v_1}}, \{k_{j_1}^{v_1}\}_{z^{v_1}},$$
$$\vdots \quad (1)$$
$$v_n, j_n, \{k_{j_n}^{v_n}\}_{y_1^{v_n}}, \cdots, \{k_{j_n}^{v_n}\}_{y_l^{v_n}}, \{k_{j_n}^{v_n}\}_{z^{v_n}} \},$$

where j_1 and j_n denote the index of the current authentication key of node v_1 and v_n, and $\{k_{j_1}^{v_1}\}_{y_1^{v_1}}$ denotes key $k_{j_1}^{v_1}$ encrypted by $y_1^{v_1}$. Note: When $K(n)$ is disseminated the first time, $j_1 = \cdots = j_n = 1$.

Receiving $K(n)$, each forwarding node decrypts the authentication keys using its own secret keys (if its secret key happens to be the same one as some secret key used to encrypt an authentication key), and stores those authentication keys it can decrypt. Then, it forwards $K(n)$ to its q *most possible next hop nodes*. These q nodes can be selected according to different metrics depending on different routing protocols used. For example, they can be those closest to the base station or having the maximum amount of energy. To save energy, $K(n)$ can be forwarded at most h_{max} hops.

We notice that the memory requirement of each node may be different, because the nodes closer to the base station are usually the hot spots and have higher memory requirement. To balance the number of keys held in each forwarding node, we propose *Hill Climbing*, in which sensor nodes do not pick y-key from a global key pool. Instead, each node picks its l y-key from l distinct hash chains of the same size. Now, a forwarding node can only decrypt an authentication key when at least one of its y-key has a larger index than the corresponding y-key of that sensing node. In *Hill Climbing*, after a forwarding node decrypts an authentication key, it encrypts that key using its own y-key and replaces the encrypted authentication key in $K(n)$ by that one its produced. Thus, the index of y-key used to encrypt authentication key increases gradually, just like climbing hill, and it is harder and harder for a downstream forward node to decrypt an authentication key. Hence, the nodes closer to the cluster stores more authentication keys of that cluster. In this way, we balance the memory requirement among the nodes by controlling the number of authentication keys a forwarding node can decrypt.

Filtering Phase: The false reports are detected and dropped by the forwarding nodes in this phase. After disseminating the authentication keys, cluster head can send the reports of some event by aggregating the messages from the sensing nodes. Each report R_i should contain t distinct MACs of those sensing nodes.

$$R_i = \{E, v_{i_1}, MAC(E, k_{kid(v_{i_1})}^{v_{i_1}}), \cdots, v_{i_t}, MAC(E, k_{kid(v_{i_t})}^{v_{i_t}})\},$$

where $v_{i_1}, \cdots, v_{i_t}$ are the indices of t sensing nodes, and $kid(v_{i_1}), \cdots, kid(v_{i_t})$ denote the indices of their current authentication keys.

As shown in Figure 1, after the cluster head hears that the next forwarding node broadcasts the reports, it sends out a message $K(t)$, which has the similar format to that of $K(n)$ besides it contains only t, but not n, encrypted authentication keys. These authentication keys are used by those sensing nodes to endorse the reports. Along the path from the cluster head to the base station, all forwarding nodes process $K(t)$ in the same way as that of the cluster head. That is, only after hearing its downstream node broadcasts the reports, can a upstream forwarding node send out $K(t)$ to this downstream node. Receiving $K(t)$, each forwarding node first decrypts the authentication keys it can. Secondly, it verifies that these decrypted authentication keys are fresher than those it stored before. Then, it verifies the MACs in the reports are produced from these decrypted authentication keys. If these two steps of verification succeed, it sends out an OK message to inform the next forwarding node to keep on forwarding the reports. Otherwise, it drops the reports and informs the next node to do so.

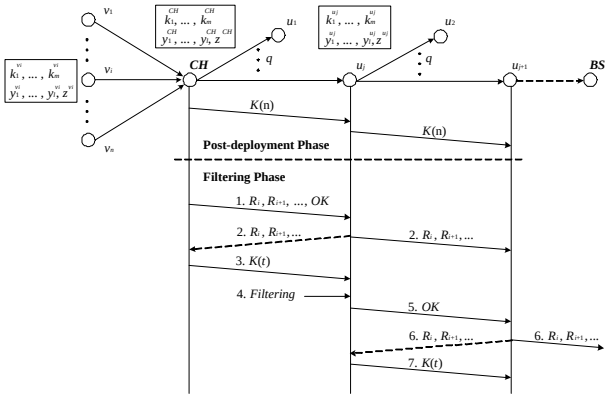

Figure 1: Detailed procedures of our scheme

4. ANALYTICAL AND SIMULATION RESULTS

We studied the detecting probability and energy savings of our scheme. The detecting probability is referred to as the probability that a node finds at least one forged MAC. Our scheme offers higher detecting probability to sensor networks than others. For example, our scheme offers a detecting probability 0.275 when SEF gives 0.05 under the same condition. If using *Hill Climbing* in our scheme, the first sensor node along the path from the cluster to the base station has a detecting probability as high as 0.775. The analytical analysis also shows that our scheme can save more energy than not using any filtering scheme or SEF under the false report attacks.

We also compared the performance of our scheme with that of others by simulation. The simulation results show that our scheme can drop the false reports earlier even with a lower memory requirement. For example, in some scenario our scheme drops the false reports within 6 hops when only storing 25 keys, but other scheme allows the false reports to travel more than 12 hops even with 50 keys stored. We simulated the dynamic topology of sensor networks by allowing sensor nodes switch their status between ON and OFF. It is shown that compared with others, our scheme has a higher filtering capacity and allows the less fraction of the false reports to reach the base station when the topology of sensor networks is changing. Moreover, our simulation indicated that *Hill Climbing* improves the filtering capacity of our scheme greatly and balances the memory requirement of sensor nodes.

We discussed several attacks specific to our scheme and the corresponding countermeasures. From the analytical and simulation studies, we concluded that our scheme outperforms others by achieving higher filtering capacity and better dealing with the dynamic topology of sensor networks.