Poster Abstract: Statistical En-route Filtering in Large Scale Sensor Networks*

Fan Ye, Haiyun Luo, Songwu Lu, Lixia Zhang
UCLA Computer Science Department
Los Angeles, CA 900095-1596
{yefan,hluo,slu,lixia}@cs.ucla.edu

Categories and Subject Descriptors

C.2.1 [**Computer Systems Organization**]: Computer-Communication Networks—*Network Architecture and Design: Wireless Communication*

General Terms

Security, Design

Keywords

compromised nodes, false data, security, sensor networks

1. INTRODUCTION

In large-scale sensor networks serving mission-critical applications, one possible denial-of-service attack is false data reports injected by attackers. Such false reports could lead to false alarms, exhaustion of the en-route sensors' limited battery energy, and congestion of wireless channels with limited bandwidth. Although recent work on sensor message authentication [2, 1] can effectively block out false data injections from external nodes, they are rendered ineffective by compromised nodes which can authenticate themselves to neighbors and correctly encrypt false messages. Such attacks through compromised nodes are possible because sensor networks are usually un-attended. An attacker can physically capture a sensor and obtain the security information stored in it without being detected.

We propose Statistical En-route Filtering (SEF) that can filter out such false reports en-route as they are forwarded toward the data collection point (called "sink"). SEF leverages the scale of the sensor network and high density level in sensor node deployment. In order to differentiate false data reports injected by compromised nodes, SEF relies on the collective efforts of both the sensors surrounding the report generation locations, and the sensors along data delivery paths.

Specifically, when an actual sensing target (called "stimulus") occurs in the field, SEF lets multiple surrounding sensors collectively generate a legitimate report that carries multiple keyed message authentication codes (MACs). These MACs are the "passport" for the report as it traverses the sensor network towards the sink. A report with less than a threshold number of MACs will be dropped. Through proper key assignments each node can only generate one legitimate MAC for each report. An attacker who captured a small number of sensor nodes has to forge incorrect MACs to inject a seemingly legitimate report.

SEF lets sensor nodes share keys probabilistically to enable statistical en-route verification of a report's MACs. Any forwarding node has certain probability of possessing one of the keys used in generating these MACs. A data report is dropped immediately upon the detection of any incorrect MAC. As more and more intermediate sensor nodes forward a data report, the probability of detecting incorrect MACs increases. Finally the sink verifies all the MACs of each received data report and filters out those false reports that escape the statistical en-route filtering.

SEF only uses computationally efficient one-way hash functions to conserve the computation resources of small sensor nodes. To minimize the communication overhead and the corresponding energy consumption, we uses Bloom filter to compress the MACs while retaining en-route verification of the MACs. Through analysis and extensive simulations, we show that with an overhead of 14 bytes per report, SEF is able to drop 80∼90% false reports injected through a compromised node within 10 hops.

2. DESIGN

SEF consists of three pieces: 1) key assignment to sensor nodes for MAC generation; 2) en-route verification of MACs to filter false data reports; 3) sink verification of each MAC to detect false reports that escape the en-route filtering.

2.1 Key Assignment and Report Generation

Nodes share keys to a certain degree to enable en-route verification of MACs; but the sharing is also constrained to prevent one compromised node from generating all the MACs required in a legitimate report.

To this end, we use a global key pool of N keys, divided into n non-overlapping partitions with $m = N/n$ keys each. Each key has a unique key index. Before a sensor node is deployed, we load it with k ($k < m$) keys and their indices, randomly chosen from one of the n partitions. That is, a sensor node only possesses keys from one single partition, but two nodes have certain probability of sharing keys because they may pick keys from the same partition. Only the sink knows all the keys.

When a stimulus appears, multiple nodes that detect it collaborate to process the signal and elect a Center-of-Stimulus

*This work is supported by DARPA under contract DABT63-99-1-0010

SenSys'03, November 5–7, 2003, Los Angeles, California, USA.
ACM 1-58113-707-9/03/0011.

node that summarizes the results and generates a report R on behalf of the group. Each detecting node then randomly selects one of its k keys K_i and generates a keyed MAC M_i for the report.

The node then sends $\{i, M_i\}$, i.e., the key index and the MAC, to the CoS. The CoS collects all the $\{i, M_i\}$'s from detecting nodes and classifies MACs based on the key partitions. We define MACs generated by keys of one key partition as one *category*. Suppose CoS collects T categories (T is a system design parameter and $T \leq n$). From each category, the CoS randomly chooses one $\{key\ index, MAC\}$ tuple and attaches it to the final report:

$$\{R, i_1, M_{i1}, i_2, M_{i2}, \cdots, i_T, M_{i_T}\}.$$

An attacker that does not compromise keys in enough number (T) of distinct partitions has to forge MACs so that its injected false data reports can be forwarded. We next describe the en-route filtering to detect and drop such false data reports.

2.2 En-route Filtering and Sink Verification

These multiple MACs collectively act as the "passport" for a report. As explained earlier, any forwarding node has certain probability of possessing one of the T keys used in generating the T MACs, thus verifying the correctness of one MAC. This probability increases as the report is forwarded over more hops.

Specifically, when a forwarding node receives a reports, it checks if there are T key indices of distinct partitions and T MACs. If not, it drops the report. Next, if it has one of the T keys (it can have at most one because each key is from a distinct partition), it recomputes the MAC using its own key. The report is dropped if the attached MAC differs from the recomputed one, which implies the MAC is forged. The report is forwarded to next hop if the attached one is the same, or this node does not have any of the T keys.

The sink does similar checking, except that it verifies all the T MACs because it knows all the keys. Any forged MAC that escapes the en-route filtering will be caught and the report rejected by the sink.

We apply Bloom filter to further compress the T MACs into a much short bit string to reduce the extra space needed in a packet. Compared to a list of MACs, the Bloom filter can reduce the overhead by 70%, to 14 bytes per report.

3. PERFORMANCE EVALUATION

SEF can detect false data reports forged by an attacker that compromises keys in up to a threshold of $T-1$ distinct partitions. As long as one forged MAC exists, the false report will be filtered en-route with increasing probability; if it escapes the en-route filtering, it will be detected at the sink. We evaluate SEF's performance against attackers compromising keys in $N_c (N_c < T)$ distinct partitions.

Figure 1 draws the portion of dropped false reports as a function of the number of hops it travels in a typical configuration. 90% false reports are dropped within 10 hops if the attacker has keys in one partition. In the worst case where only one forged MAC exists, 80% are dropped in 32 hops and they travel 20 hops on average.

The attacker might correctly "guess" a MAC and sets the Bloom filter bits. We calculate such probabilities and find that they are so small that their effect on en-route filtering is negligible. Even in the worst case of one forged MAC, the attacker has only 2^{-20} probability of forging it correctly and cheating the sink successfully.

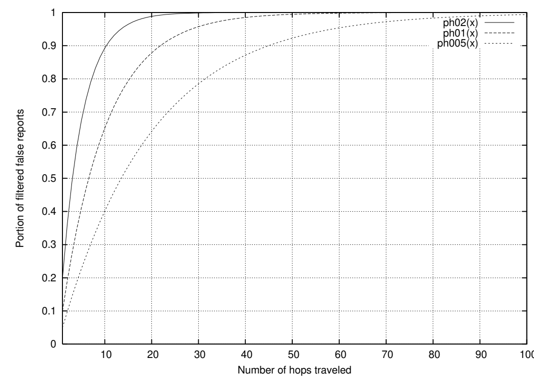

Figure 1: The portion of dropped false reports as a function of the number of traveled hops. The three curves are for $N_c = 1, 3, 4$, $T = 5$.

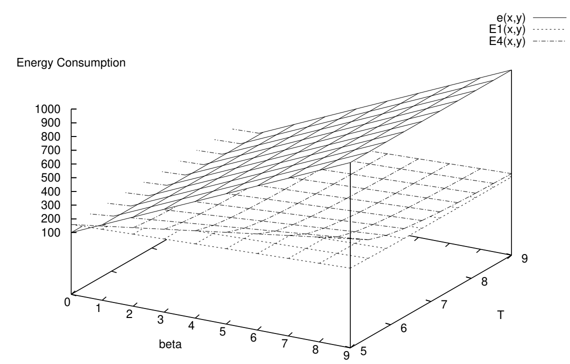

Figure 2: e is the energy amount without SEF; $E1, E4$ are the amounts with SEF for $4, 1$ forged MAC(s) per report.

We compare energy consumption with and without SEF and find that SEF saves energy when the amount of injected traffic exceeds that of the legitimate traffic. For example, even in the worst case where only one MAC is forged, about 67% less energy is consumed. In reality, an attacker can inject traffic in amounts magnitudes higher than that of legitimate traffic. SEF saves large amounts of energy in such cases.

We plan to further exploit the dense deployment to detect reports forged by attackers compromising keys in T or more, but less than n distinct partitions. We are also considering exploiting location information to eliminate that threshold.

4. REFERENCES

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

[2] V. Wen, A. Perrig, and R. Szewczyk. SPINS: Security Suite for Sensor Networks. In *ACM MOIBCOM*, 2001.