Süleyman Demirel Üniversitesi,
Fen Bilimleri Enstitüsü Dergisi,
11-3 (2007),239-245

False Data Detection in Wireless Sensor Networks via Merkle Hash Trees

Suat ÖZDEMİR

Gazi Üniversitesi, Mühendislik-Mimarlık Fakültesi, Bilgisayar Mühendisligi Bölümü-Maltepe / ANKARA
Received:03.08.2007, Accepted:24.01.2008

Abstract: Large-scale wireless sensor networks provide economical and viable solutions to many monitoring and tracking problems. However, practical deployment of wireless sensor networks introduces problems that do not occur in traditional networks. For example, individual sensor nodes are prone to security compromises. Moreover, compromised nodes can inject false sensing reports into the network. If undetected, false data injection attacks may deceive the base station and deplete the limited energy resources of relaying sensor nodes. Standard data authentication schemes cannot prevent these attacks if there exists more than one compromised node in the network. In this paper, we present a collaborative data authentication protocol that detects false data injection attacks. In the proposed protocol, false data injected by less than *n* compromised nodes are detected and eliminated by constructing consecutive Merkle Hash Trees (MHT) between the source node and the base station. The security and performance analysis show that the proposed protocol is able to detect false data injected by less than *n* compromised nodes and incurs less resource consumption compared to previous false data detection schemes.

Key Words: Security, Sensor Networks, False Data Detection, Merkle Hash Trees

Kablosuz Algılayıcı Ağlarında Merkle Özet Ağaçları ile Yanlış Veri Bulunması

Özet: Büyük ölçekli kablosuz algılayıcı ağları birçok izleme ve takip problemine uygun ve ekonomik çözümler getirmiştir. Ancak, bu ağların çabuk ve kolay kurulabilmeleri geleneksel ağlarda görülmeyen bir takım problemleri ortaya çıkarmıştır. Örneğin, algıyacılar ele geçirme ataklarına karşı korunmasızdırlar. Dahası, ele geçirilmiş algılayıcılar ağa yanlış veri gönderebilirler. Eğer önlenmez ise, bu yanlış veri gönderme atakları baz istasyonunu yanıltabilir ve veri aktarımı yapan algılayıcıların kaynaklarını tüketebilir. Eğer ağda birden fazla ele geçirilmiş algılayıcı varsa, standart veri doğrulama metodları bu atakları engelleyemez. Bu çalışmada, yanlış verileri bulan bir iş birlikçi veri doğrulama protokolü sunulmuştur. Sunulan protokolde *n*'den az sayıda ele geçirilmiş algılayıcı tarafından gönderilen yanlış bilgi kaynak algılayıcı ile baz istasyonu arasında ardışık Merkle özet ağaçları kurularak bulunur. Güvenlik ve performans analizi, önerilen protokolün *n*'den az sayıda ele geçirilmiş algılayıcı tarafından gönderilen yanlış verileri bulduğunu ve daha önce geliştirilmiş yanlış veri bulma protokollerine göre daha az kaynak kullanımına neden olduğunu göstermiştir.

Anahtar Kelimeler: Güvenlik, Algılayıcı Ağları, Yanlış Veri Bulma, Merkle Özet Ağaçları

Introduction

Wireless sensor networks are usually deployed in unattended hostile environments to monitor target regions by employing large number of sensor nodes with limited wireless communication, computation and sensing capabilities (Akyıldız *et al*., 2002). Due to their unattended nature, network security is an essential requirement for mission-critical wireless sensor network applications, such as border surveillance or battlefield monitoring. The reason is that non-tamper proof sensor nodes are prone to node compromise attack in which intruders gain the control of sensor nodes. Such compromised nodes inject false data into the network in order to mislead the base station. In addition to deceiving the base station, forwarding false data depletes the constrained resources of sensor nodes such as battery power and bandwidth. Therefore, false data injected by compromised nodes must be detected and eliminated as early as possible to minimize the resource consumption and to gather correct information at the base station.

In response to false data injection attacks, we introduce a collaborative data authentication protocol, called Merkle Hash Tree Authentication (MHTA). Protocol MHTA aims to defend against false data injection attacks launched by up to *n*-1 compromised nodes, where *n* is a system parameter that changes depending on the security needs of the application and the node density of the network. The basic idea behind MHTA is to divide sensor nodes into groups so that each node group detects false data by collaboratively constructing a Merkle Hash Tree (MHT) over the forwarded data. Each node of an MHT represents a hash value of the forwarded data computed by a group member. The number of leaf nodes on an MHT is equal to system parameter *n*. Therefore, forwarding nodes on each path to the base station are divided in groups of size *n*. i^{th} group from the source node is represented by G_i. Each group G_i collaboratively constructs two MHTs, namely MHT_v^i to check if any false data is injected while data is forwarded by group G_{i-1}, and MHT_a^i that will be verified by group G_{i+1}. If any part of the data is changed while data is forwarded in group G_i, the false data is detected during the construction of MHT_v^{i+1} in the group G_{i+1}. Figure 1 depicts an example authentication and verification process of protocol MHTA. Due to early detection and elimination of false

suatozdemir@gazi.edu.tr

data, the protocol MHTA not only ensures the correct data delivery to the base station but also improves the energy efficiency of the network significantly. The performance analysis and simulation results show that, considering the security it provides, protocol MHTA is energy efficient and incurs less communication overhead compared to previous false data detection techniques.

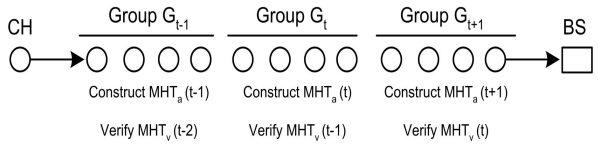

Figure 1. *False data detection via collaborative data authentication.*

The remainder of the paper is organized as follows. We first give the related work and then system and threat models. After that MHTA protocol is presented in detail. Last two sections present the performance analysis of MHTA and concluding remarks, respectively.

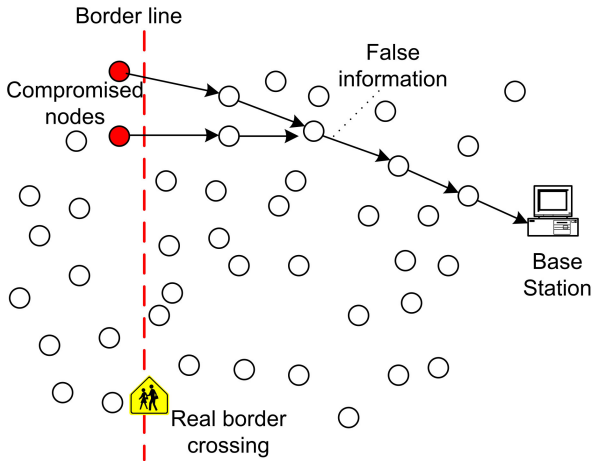

Figure 2. *Compromised nodes inject false data about a fake border crossing to mislead border patrol.*

Related Work

Consider the scenario presented in Figure 2 where compromised nodes inject false data about a fake border crossing in order to deplete the energy of sensor nodes and to mislead border patrol. In general, it is not possible to prevent the injection of false data because sensor nodes are vulnerable to node compromise attacks. However, it is possible to detect and drop false data early using Message Authentication Codes (MACs) in data authentication schemes (Ye *et al.,* 2004, Zhu *et al.,* 2004, Yang and Lu, 2004, Yu and Guan, 2006). The statistical en-route detection scheme (Ye *et al.,* 2004), called SEF, enables relaying nodes and the base station to detect false data with a certain probability. To detect and filter out forged messages, SEF relies on the collective decisions of multiple sensor nodes. When an event occurs in an area of interest, the surrounding sensor nodes generate a legitimate report that carries multiple MACs and forwarding nodes detect incorrect MACs and filter out false reports with some probability.

The interleaved hop-by-hop authentication scheme (Zhu *et al.,* 2004) guarantees that the base station detects any packet containing false data if at least t +1 sensor nodes agree upon a report when there are at most t colluding compromised nodes. The value of a security threshold t is determined based on the network node density and the security requirements of the application under consideration. All the nodes that are involved in forwarding a report to the base station authenticate the report in an interleaved hop-by-hop fashion.

The Commutative Cipher based En-route Filtering scheme (CCEF) (Yang and Lu, 2004) drops false data en-route using the public key cryptography instead of sharing symmetric keys. 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. The base station first prepares two keys, namely session and witness keys, for each source node per session. Then, the base station sends these keys to the respective source nodes. Although session keys are encrypted prior to their transmission to source nodes, the witness keys are transmitted in plaintext so that all intermediate nodes between the base station and source nodes can also use them later for data verification. A forwarding intermediate node can use the witness key for data verification, and drops the false data whenever it is detected.

In the dynamic en-route filtering scheme (Yu and Guan, 2006), each legitimate data are endorsed by multiple sensor 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. The hill climbing approach guarantees that the nodes closer to a cluster head stores more authentication keys than those nodes farther from the cluster head. This leads the number of authentication keys stored in each forwarding node to be balanced. If a forwarding node has the authentication key, it can validate the authenticity of the reports and drop the false data.

System Model

Assumptions

We consider a static cluster based wireless sensor network consisting of a large number of sensor nodes. Each cluster has a controller node called cluster head which is responsible for data aggregation and communication with the base station. Sensor nodes take turns to be the cluster head, however forming clusters and selection of the cluster heads are out of scope of this paper. Data collection is done at a powerful base station located nearby the sensor network. Cluster heads collect data from their cluster members, aggregate the data, and forward it to the base station over multi hop paths. In order to prevent a cluster head from manipulating data during data aggregation process, at least n-1 sensor nodes

monitor the cluster head using a monitoring technique (Marti *et al.,* 2000, Ganeriwal and Srivastava, 2004). Sensor nodes are battery powered and composed of a small computation unit, a sensing unit and a radio. For example MICA2 motes (Crossbow Tech., 2007) have a 4-Mhz, 8-bit Atmel microprocessor, and are equipped with 128KB of instruction memory and 4KB of RAM. We assume that the amount of memory is enough for storing security keys and data required for monitoring. We also assume that sensor nodes establish shared keys with their neighbors and the base station using one of the existing key establishment schemes (Du *et al.,* 2003, Liu and Ning, 2003, Eschenauer and Gligor, 2002, Ozdemir and Cam, 2006) which are shown to be affordable for sensor networks in terms of computation and communication overhead.

Threat Model

In this paper, we focus on false data injection attacks in which compromised nodes collude to deceive the base station and/or to deplete the constrained resources of forwarding nodes. We assume that the attacker may know the security mechanisms that are deployed in the sensor network. Intruders can compromise any sensor node in the network but the base station. Besides false data injection, compromised nodes can perform various other attacks such as jamming, DoS, replay attacks or injecting false control packets. However, in this paper, we do not address such attacks and focus only on the false data injection attack.

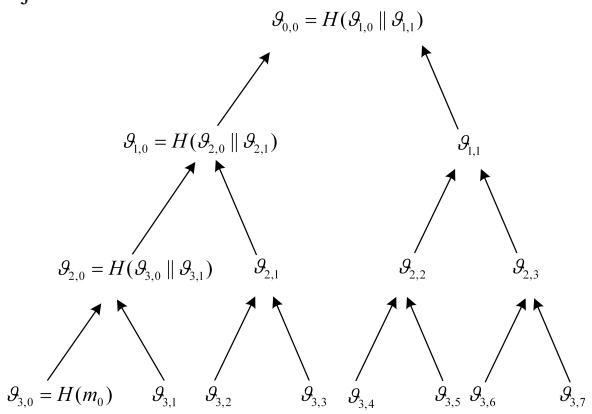

Figure 3. *Sample Merkle hash tree. $H(m_0)$ represents hash of message m_0. Each ϑ is result of its hash of its two children values. If any node of tree is changed, ϑ value at the root changes.*

MHTA: Merkle Hash Tree Authentication Protocol

Before presenting the details of MHTA protocol, we give some introductory information about Merkle Hash Trees (MHT) (Merkle, 1980). Basically, an MHT is a binary tree in which the root of the tree is a commitment to its leaves. As presented in Figure 3, each internal value of the tree is computed by its two children $\vartheta_{i,j} = H(\vartheta_{i+1,2j} \| \vartheta_{i+1,2j+1})$. $H(\cdot)$ is a one way hash function and $\|$ represents the concatenation operation. The root node can verify any leaf value by traversing the tree. Due to page limitations, we are not able to give detailed information about Merkle hash trees but interested readers are referred to (Merkle, 1980). In this paper, we employ MHTs for collaborative data authentication and verification. As explained in subsequent sections, rather than using a single node to construct the MHT, in protocol MHTA, forwarding sensor nodes collaboratively construct the tree and the root of the MHT is used to verify the integrity of the forwarded data. To construct MHTs for data authentication, sensor nodes located on the paths to the base station form groups and establish associations among themselves.

Path and Association Establishment

In MHTA protocol, each sensor node N_i on a path from any cluster head (CH) to the base station (BS) has a pair of cooperating nodes called lower cooperating node (LC) and upper cooperating node (UC) which are n hops away from N_i. The LC of N_i is located between CH and N_i whereas UC of N_i is located between N_i and BS. If the distance between N_i and BS is less than n hops then BS becomes UC of N_i. An example system model is shown in Figure 4.

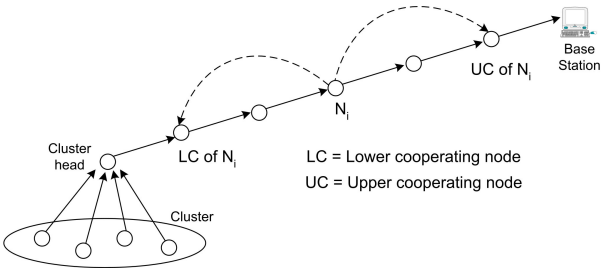

Figure 4. *Graphical presentation of the system model (n=2).*

In order to perform protocol MHTA, relaying nodes must discover their UC and LC nodes and establish secret keys with each of them. Once nodes discover their UC and LC nodes, then each sensor node is responsible for maintaining its own UC and LC using the route maintenance technique introduced in (Zhu *et al.,* 2004). BS starts UC and LC discovery phase by broadcasting an UC discovery message, called M_{UC}. M_{UC} is recursively forwarded until it reaches all cluster heads in the network. On receiving an M_{UC} message, each sensor node appends its node ID to the end of M_{UC} until there is a list of n node IDs. If there is already n node IDs in the list, the sensor node deletes the first node ID and appends its own ID to the end of the list. Hence, regardless of the number of hops an M_{UC} travels, the number of IDs appended to that M_{UC} is limited by n. Each sensor node store the first node ID of the list appended to M_{UC} as its UC node. CH stores all the IDs in the list rather than storing only the first one, then it assigns each one of those n IDs to one of its monitoring cluster nodes as a UC node.

CH replies the received M_{UC} message with an LC discovery message, called M_{LC}, which includes the ID of CH and the IDs of cluster nodes that are assigned UC nodes. M_{LC} follows its associated M_{UC}'s path in reverse direction and reaches BS. Relay nodes discover their LC

nodes in M_{LC} forwarding phase. As opposed to M_{UC} forwarding phase, on receiving M_{LC} each sensor node deletes the last ID from the list and appends its own ID as the first ID of the list. Each node stores the last node ID of the list as its LC node. Relay nodes that receive data from multiple cluster heads have multiple LC nodes and those paths are identified by unique cluster IDs. Once all sensor nodes discover their UC and LC nodes, each node establishes pairwise secret keys with its UC and LC nodes. We denote the pairwise shared key between node N_a and node N_b as $K_{a,b}$ where $K_{a,b} = K_{b,a}$.

Collaborative Authentication via Merkle Hash Trees

Data collected in a cluster aggregated by CH and relayed to BS. Protocol MHTA guarantees the integrity of the aggregated data (D_{agg}) by constructing consecutive MHTs during data forwarding. Depending on its location, each forwarding node computes the hash of D_{agg}'s MAC and performs some other hash operations needed to construct the MHT.

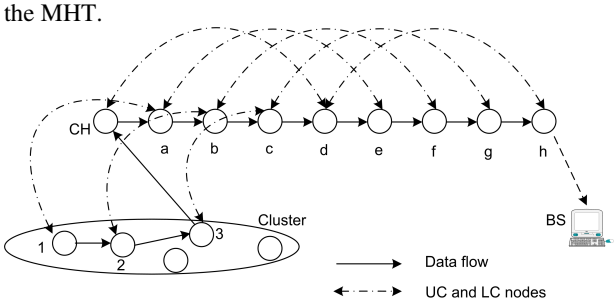

Figure 5. *MHT construction example (n = 4). Dashed lines show the UC-LC relationships.*

Consider Figure 5 where the network parameter $n = 4$. As explained previously, let us assume that cluster nodes N_1, N_2 and N_3 monitor CH. Upon collection of cluster data, CH aggregates it, and broadcasts the aggregated data D_{agg} along with the IDs of the monitoring nodes and its own cluster head ID (N_1, N_2, N_3, CH). Monitoring nodes verify the correctness of $D_{agg.}$ In order to authenticate D_{agg}, at least (n-1) monitoring sensor nodes and CH must construct the first *authentication* MHT (MHT_a^0) tree over D_{agg}. If there is no false data injection into D_{agg}, each monitoring node N_j computes a leaf of MHT_a^0 ($\vartheta_{i,j}$, $i=log_2 n$) by hashing MAC(D_{agg}) value computed using the pairwise key that N_j shares with its UC node. CH computes the right most leaf of the MHT_a^0 ($\vartheta_{i,n}$, $i=log_2 n$). Each sensor node forwards the leaf it computes to the next sensor node on the path. When a node has two children of a parent (for example $\vartheta_{3,2}$ and $\vartheta_{3,3}$ on Figure 3), it computes the parent of those children ($\vartheta_{2,1}$) and forwards the parent value to the next node. Figure 6 shows the construction of the initial MHT_a^0 in the cluster presented in Figure 5. Then, we explain the construction of the initial MHT_a^0 in detail. Note that $MAC_{K1,a}(D_{agg})$ represents the MAC of D_{agg} computed using the shared key between N_1 and $N_{a.}$

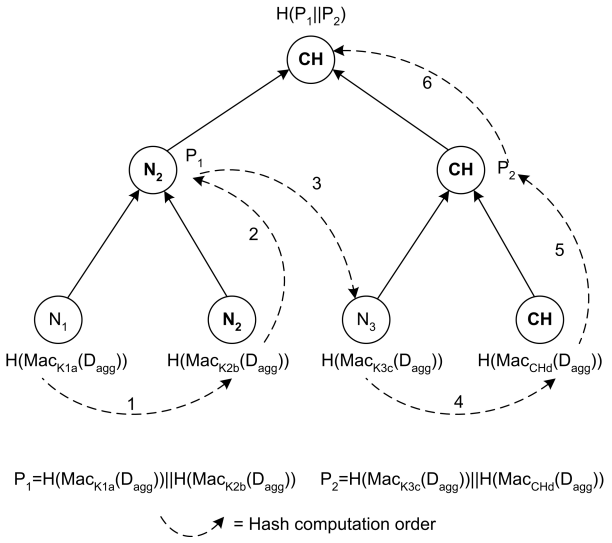

Figure 6. *Stepwise representation of the initial MHT construction by N_1, N_2, N_3, and CH.*

Sensor node N_1 computes $H(MAC_{K1,a}(D_{agg}))$ and sends the following message to N_2

$$N_1 \rightarrow N_2: D_{agg}, H(MAC_{K1,a}(D_{agg}))$$

Sensor node N_2 receives $D_{agg}, H(MAC_{K1,a}(D_{agg}))$ from sensor node N_1 and computes $H(MAC_{K2,b}(D_{agg}))$. Since $H(MAC_{K1,a}(D_{agg}))$ and $H(MAC_{K2,b}(D_{agg}))$ are the children of the same parent, N_2 also computes $H(H(MAC_{K1,a}(D_{agg})) \parallel H(MAC_{K2,b}(D_{agg})))$. Let us call this value P_1. N_2 sends the following message to N3.

$$N_2 \rightarrow N_3 : D_{agg}, P_1$$

Sensor node N_3 receives P_1 value from N_2, computes $H(MAC_{K3,c}(D_{agg}))$, and sends the following to CH

$$N_3 \rightarrow CH: D_{agg}, P_1, H(MAC_{K3,c}(D_{agg}))$$

CH receives P_1 and $H(MAC_{K3,c}(D_{agg}))$ from N_3 and computes $H(MAC_{KCH,d}(D_{agg}))$. Since $H(MAC_{K3,c}(D_{agg}))$ and $H(MAC_{KCH,d}(D_{agg}))$ are the children of the same parent, CH also computes the following.

$$H(H(MAC_{K3,c}(D_{agg})) \parallel H(MAC_{KCH,d}(D_{agg}))).$$

Let us call this value as P_2. Note that CH also has P_1. Since P_1 and P_2 are the children of the same parent, CH also computes $H(P_1||P_2)$ which we call as P_3. Hence, the following message is sent to N_a from CH.

$$CH \rightarrow N_a : D_{agg}, P_3, P_2, P_1 \text{ where } P_3 = H(P_1||P_2)$$

Since n=4, P_3 is the root of MHT_a^0 and it is a commitment to its leaves which are indeed the MACs of D_{agg}. Therefore, N_a, N_b, N_c, and N_d in Figure 5 will use P_3 to verify D_{agg}. To achive the verification nodes N_a, N_b, N_c, and N_d constructs the verification MHT (MHT_v^0) using the same keys that their LC nodes (N_1, N_2, N_3 and CH) used. While MHT_v^0 is being constructed by N_a, N_b, N_c, and N_d, they compare MHT_v^0's P_1 P_2 and P_3 values with P_1 P_2 and P_3 values of MHT_a^0. If all values match then the authentication of D_{agg} is verified. However, if N_a, N_b, N_c, and N_d compute a P value for MHT_v^0 which is different than MHT_a^0's respective P value, then D_{aag} is dropped immediately. Moreover, N_a, N_b, N_c, and N_d also construct MHT_a^1 using the keys that they share with their UC nodes (N_e, N_f, N_g, and N_h, respectively). MHT_a^1 is verified by the nodes N_e, N_f, N_g, and N_h by constructing MHT_v^1. Hence, each constructed MHT_a by n nodes is verified by the next n nodes via constructing the corresponding MHT_v. This interleaved process continues until D_{agg} reaches the base station resulting in authenticated data delivery. Note that if there are n compromised nodes in the network; they are able to inject false data. Therefore, protocol MHTA can prevent false data injection attacks up to n-1 compromised nodes.

Performance Analysis and Results

In this section, we analyze the security of protocol MHTA and evaluate its computation and communication performance, respectively.

Security Analysis

The MHTA security analysis can be divided into two parts, inside the cluster and on the path from the cluster head to the base station. For inside cluster attack, all the n-1 nodes and CH must be compromised nodes in order to inject false data. If this is not the case, there will be at least one noncompromised who participate in construction of MHT and injected false data will be detected by the noncompromised node's UC node. Here the assumption is that noncompromised nodes must have the correct aggregated data using monitoring techniques. Monitoring schemes proposed in (Marti *et al.,* 2000, Ganeriwal and Srivastava, 2004) are shown to be effective in detection of compromised nodes and can be employed along with protocol MHTA. In addition, false data injections during data forwarding are also detected by protocol MHTA if there exists less than n compromised node on the path.

Lemma 1: False data injected by less than n compromised node during data forwarding is dropped by MHTA protocol.

Proof: An intruder who compromises only one node can replace forwarded data with false data and compute a hash value over the false data. Since an MHT is constructed using n hash values, in order to construct an MHT over a false data, the intruder must compromise n consecutive nodes on the path to base station. Since each MHT_a is verified by its corresponding MHT_v that is constructed by the next node group, a false data packet is forwarded if and only if those corresponding MHT_a and MHT_v match. Therefore, if there is a non-compromised node among any n consecutive forwarding nodes on the path, false data will be dropped due to unmatched MHT_a and MHT_v pair.

Computation and Communication Analysis

The computation overhead of MHTA is mainly due to the MACs that are computed to construct MHTs. In MHTA, to construct an MHT n MAC and 2n-1 hash computations are required. Since for each forwarded data two MHTs must be constructed by n sensor nodes, each sensor node performs 2 MAC and 4 hash computations on average. The energy required for hash computations is negligible as the size of the MACs is around 4 bytes. Hence, compared to previous approaches (Ye *et al.,* 2004, Zhu *et al.,* 2004, Yang and Lu, 2004, Yu and Guan, 2006), protocol MHTA's energy consumption due to MAC computation is significantly reduced. For example, in the scheme proposed in (Zhu *et al.,* 2004), the average number of computed MACs in each sensor node is 4. Although, this is a very good improvement, computation is not the primary bottleneck in wireless sensor networks. In (Ye *et al.,* 2004) it is shown that computing a MAC is equivalent to transmitting one byte, therefore computation is not the limiting factor for protocol MHTA but communication.

Communication cost of MHTA is due to following reasons; (i) establishing secret keys among sensor nodes, (ii) transmitting the hash values to construct MHTs. In MHTA, any key distribution scheme that allows sensor nodes to establish secret keys with multihop neighbors can be used. There are number of key establishment schemes that are shown to be feasible for wireless sensor networks (Du *et al.,* 2003, Liu and Ning, 2003, Eschenauer and Gligor, 2002, Ozdemir and Cam, 2006). Since we do not propose any new key establishment protocol, we will not analyze the communication overhead due to key establishment. But we will consider the communication overhead due to transmitted hash values. For example, in MHTA and the scheme proposed in (Zhu *et al.,* 2004), the number of hash values added to each message depends on the network parameter n. However, in the scheme proposed in (Zhu *et al.,* 2004) each data packet is accompanied by n+1 regular size MACs whereas in MHTA at most n-1 hash values of MACs (n/2 on average) are transmitted with a data packet. In the next subsection, we present simulation results for these two schemes.

Simulation Results

The communication performance of a wireless sensor network using MHTA is evaluated in comparison with the network using the false data detection scheme proposed in (Zhu *et al.,* 2004). Simulations are performed with connected random network instances generated in the target region using QualNet (Qualnet, 2007) a PARSEC based commercial sensor network simulator. Due to poor

radio conditions of sensor networks, a retransmission mechanism is implemented in the simulations and the retransmission limit is set to 5. The default channel error rate is 10% which is usually accepted as a poor radio condition. Results are averaged over 20 network instances with 100 sensor nodes and various numbers of compromised nodes (1 to 8).

As seen from Figure 7, MHTA incurs less communication overhead than Zhu *et al.'s* scheme. This is due to collaborative construction of MHTs in protocol MHTA versus individual MAC authentications in the scheme introduced in (Zhu *et al.,* 2004). In protocol MHTA, whenever a node has the both children of a parent, the node calculates the parent and forwards the parent value rather than two children. Therefore, the number of MAC transmissions is reduced. Moreover, the scheme proposed in (Zhu *et al.,* 2004) transmits MACs along with the forwarded data whereas MHTA transmits the hashes of the MACs which are much smaller than actual MACs.

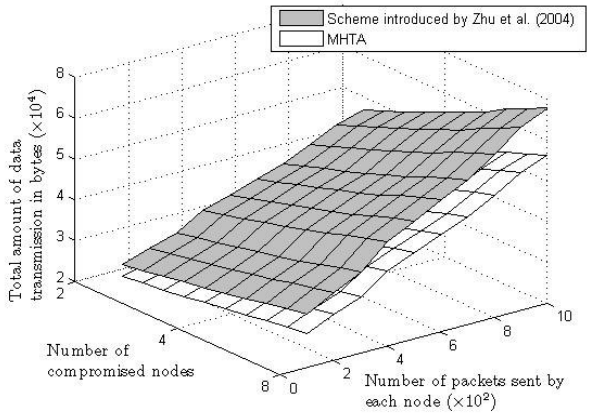

Figure 7. *Communication overhead of MHTA and false data detection scheme of Zhu et al., 2004.*

Simulations are also conducted for various SNR values to show the impact of packet loss rate. Figure 8 shows the total data transmission in the network for different SNR values for MHTA protocol. As seen from Figure 8, performance of MHTA is slightly affected by different SNR values which indicate that MHTA is suitable for noisy sensor network environments.

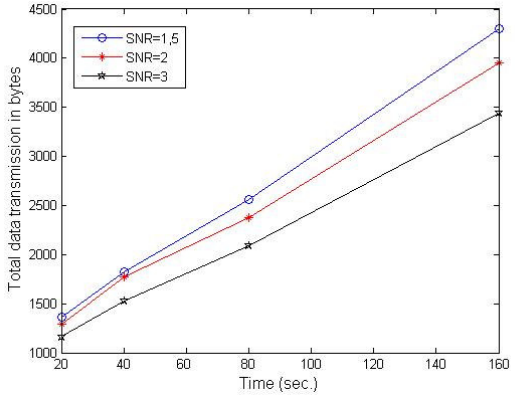

Figure 8. *The total data transmission in the network versus various SNR values.*

Finally, the impact of *n* on the communication overhead is evaluated. The value of *n* affects the number of hash value transmissions between forwarding nodes. Therefore, as it is also shown in the communication analysis, increasing the value of *n* increases the total data transmission in the network. The simulation is carried out for *n* values between 2 and 16 and results are presented in Figure 9 which indicates that increasing *n* form 2 to 16 results in 57% increase in data transmission. However, when *n*=16 the provided security is much higher than the case *n*=2. therefore, an intruder must compromise 16 nodes to inject false data into the network. Hence, the tradeoff between security and data communication overhead must be balanced by considering the requirements of the sensor network application.

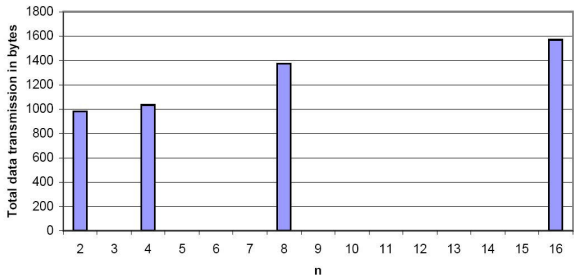

Figure 9. *The impact of n on the communication overhead.*

Conclusion

In wireless sensor networks, compromised sensor nodes can inject false data to deceive the base station and/or to deplete the already-constrained resources of forwarding nodes. In this paper, we presented Merkle Hash Tree Authentication (MHTA) protocol that detects false data injections before the false data reaches the base station. In protocol MHTA, sensor nodes collaboratively authenticate the forwarded data by constructing Merkle Hash Trees (MHT) over the forwarded data and the false data is detected by verifying those MHTs. MHTA protocol is compared with the previous false data detection schemes using several simulation scenarios. Simulation results show that MHTA protocol performs better than previous techniques. Furthermore, due to early detection of the false data; MHTA protocol reduces the energy and bandwidth consumption of forwarding nodes significantly. In addition, the performance analysis of MHTA shows show that the computational and communication overhead of protocol MHTA is not substantial, thereby making the implementation of MHTA protocol feasible.

References

Akyıldız, I.F., Su W., Çayırcı, E. 2002. A survey on sensor networks. IEEE Communications Magazine 40, 102-114.

Crossbow Technologies, 2007 http://www.xbow.com/ (Erişim tarihi: 03/08/2007).

Du, W., Deng, J., Varshney, P. K. 2003. A pairwise key pre-distribution scheme for wireless sensor networks, 10th ACM Conference on Computer and Communications Security, October 27-30, Washingtion, DC, 42-51.

Eschenauer, L., Gligor, V. D. 2002. A key-management scheme for distributed sensor networks, 9th ACM conference on Computer and communications security, Washington, DC, 41-47.

Ganeriwal, S., Srivastava, M. B. 2004. Reputation-based framework for high integrity sensor networks, 2nd ACM workshop on Security of ad hoc and sensor networks, Washington DC, 66-77.

Liu, D., Ning, P. 2003. Establishing pairwise keys in distributed sensor networks, 10th ACM Conference on Computer and Communications Security, October 27-30, Washingtion, DC, 52-61.

Marti, K. L. S., Giuli, T. J., Baker, M. 2000. Mitigating routing misbehavior in mobile ad hoc networks, 6th Intl. Conference on Mobile Computing and Networking, August 6-11, Boston, Massachusetts, 255-265.

Merkle, R. C. 1980. Protocols for public key cryptosystems, IEEE Symposium on Research in Security and Privacy, 122-134.

Ozdemir, S., Cam, H. Key establishment with source coding and reconciliation for wireless sensor networks, IEEE IPCCC 2006, April 2006, Phoenix, AZ, 407-414,.

QualNet, http://www.scalable-networks.com/ (Erişim tarihi: 03/08/2007)

Yang, H., Lu, S. 2004. Commutative Cipher Based En-Route Filtering in Wireless Sensor Networks, IEEE VTC Fall 2004, Los Angeles, CA, 1223- 1227.

Ye, F., Luo, H., Lu, S., Zhang, L. 2004. Statistical En-route Detection and Filtering of Injected False Data in Sensor Networks, IEEE INFOCOM 2004, March 7-11, Hong Kong, China, 2446-2457.

Yu, Z., Guan, Y. 2006. A Dynamic En-route Scheme for Filtering False Data in Wireless Sensor Networks, IEEE INFOCOM 2006, April 23-27, Barcelona, Spain, 1-12.

Zhu, S., Setia, S., Jajodia, S., Ning, P. 2004. An Interleaved Hop-by-Hop Authentication Scheme for Filtering False Data in Sensor Networks, IEEE Symposium on Security and Privacy, Oakland, CA, 259-271.