INTERNATIONAL JOURNAL OF INTELLIGENT CONTROL AND SYSTEMS
VOL. 12, NO. 4, DECEMBER 2007, 341-350

Localized Broadcast Authentication in Large Sensor Networks

Qijun GU, Jawad DRISSI

Abstract— **Although TESLA[1], [2] can provide the crucial broadcast authentication, it also introduces large verification delay that limit its application in a large wireless sensor network. In this paper, we present two localized broadcast authentication protocols that reduce verification delay via using a few trusted nodes. These trusted nodes divide sensor nodes in a large sensor network into several subsets, and coordinate broadcast authentication in the subsets. Thereby, all sensor nodes can verify a broadcast message as if they were in a small network. One of the proposed protocols, L-TESLA, is developed to identify the subsets according to the deployment of the trusted nodes. However, the delay reduction of L-TESLA is at the expense of increasing broadcast overhead due to updating localization information. To address the problem, we propose LD-TESLA to incorporate the technique of dominating nodes in broadcast authentication. The simulation results illustrate that both protocols significantly reduce the average verification delay. At the same time, they provide tradeoff between verification delay and broadcast overhead, and thus can work as alternative broadcast authentication protocols serving different applications.**

Index Terms— **authentication, sensor networks, dominating sets**

I. Introduction

Recent advances in wireless communications and electronics have enabled the development of low-cost sensor networks. Large scale wireless sensor networks are deployed because they greatly extend our ability to monitor and control the physical environment from remote locations. Such networks can greatly improve the accuracy of information obtained via collaboration among sensor nodes and online information processing at those nodes. Wireless sensor networks improve sensing accuracy by providing distributed processing of vast quantities of sensing information (e.g., seismic data, acoustic data, high-resolution images, etc.). Sensors can aggregate such data to provide a rich, multi-dimensional view of the environment. Sensor networks can continue to function accurately in the face of failure of individual sensors. The wireless network architecture allows sensors to be rapidly deployed and without modification to large structures and systems, in a broad spectrum of applications ranging from battlefield perimeter security and shoreline reconnaissance to personnel health monitoring.

Nevertheless, many applications in sensor networks are highly dependent on security. If the network is compromised or disrupted, these applications will fail to provide desired information. In this paper, we focus on the security of broadcast that is commonly used for disseminating critical data (such as requests to monitor specific targets or network-wide keys and time information) to all sensor nodes. To ensure the integrity of broadcast messages, various broadcast authentication schemes have been proposed. Since sensor nodes work under a set of constraints arising from limited energy, computational power, and communication resources, TESLA [1], [2] presents quite a few advantages over other approaches [3], [4], [5], [6], [7]. For example, sensor nodes only need to compute light-weight hash values for verification in TESLA.

However, the least computation demand is at the expense that all sensor nodes in a network should buffer received broadcast messages until a later time when the source releases keys for them to verify packets. In a large sensor network, such a requirement can limit its application, because the verification delay may be too large for sensor nodes to allocate enough buffering storage or may reduce the timeliness of the broadcast information. To address this issue, we propose a localized broadcast authentication scheme L-TESLA which divides sensor nodes into several subsets in which they can be managed by some better secured nodes (called trusted nodes). These trusted nodes coordinate broadcast authentication in each subset so that sensor nodes can verify a broadcast message as if they were in a small network.

Nevertheless, L-TESLA increases broadcast overhead for reducing broadcast delay. When sensor nodes find closer trusted nodes, they will notify other nodes by broadcasting updated information which results in overhead increasing. At the same time, when nodes rebroadcast, many packets may be simply discarded as duplicated packets and thus do not contribute to information delivery in the network. Being aware of these two properties in broadcast, we propose another protocol LD-TESLA that integrates dominating set protocols (DOM) [8], [9] to eliminate redundant rebroadcasts in broadcast authentication. Because DOM only requires a portion of sensor nodes (called dominating nodes) to broadcast, broadcast overhead can be significantly reduced.

Contribution. This paper contributes to the literature in three aspects. First, the proposed broadcast authentication protocols take full advantage of roles of network nodes and local information in a sensor network, in contrast to many existing broadcast authentication schemes. Both broadcast authentication and TESLA key chains are localized. Second, the proposed protocols can be easily adopted in sensor networks. Especially in a large sensor network, sensors are organized in a non-flat structure, such as clustered [10], [11] or hierarchical [12], [13]. Managing nodes (e.g. headers of clusters or nodes in high levels of a hierarchy) are usually more powerful than other nodes and thus can be better secured and function as trusted nodes. Third, L-TESLA and LD-TESLA provide alternative solutions regarding tradeoff between verification delay and broadcast overhead for satisfying applications with different requirements.

Qijun Gu and Jawad Drissi are with Department of Computer Science, Texas State University at San Marcos, San Marcos, TX, 78666. Emails are {qg11,jd30}@txstate.edu

The remainder of the paper is organized as follows. Section II presents related works. Section III presents the subject problem, the notations and the initialization in our protocol. Section IV and V explain the proposed solutions. Evaluation of the proposed protocols is presented in Section VI, including analysis on security and performance. Finally, we conclude the paper and present future research directions in Section VII.

II. Related Works

The proposed protocols are based on recent advances in broadcast and authentication techniques. In the following, we present the related works in these two areas.

A. Broadcast Authentication

Broadcast authentication allows receivers of broadcast data to verify that the received data was originated from the claimed source and was not modified in route. Many group authentication approaches [5], [6], [7], [14], [15], [16], [17] have been proposed to amortize the cost of a digital signature over multiple packets. However, they normally demand computation power beyond a sensor's capability. They either require receivers to verify all the packets in a batch or to verify every packet individually with the computational cost of a single digital signature for every block of packets. They also suffer from packets loss or high packet overhead. Researchers also studied problems in authentication, including losses of signature packets [18], and injection of false data packets into the transmission channels [3], [4].

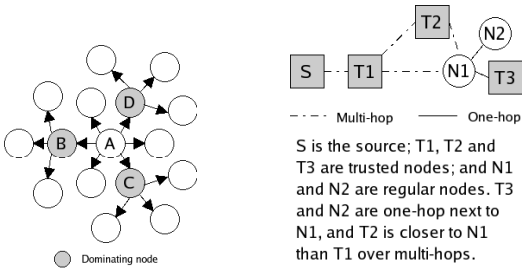
Perrig proposed an efficient and practical broadcast authentication approach, TESLA [2], which mainly uses symmetric cryptography for authentication, and only requires loose time synchronization between a sender and multiple receivers. Due to its advantages, we will take the basic idea of TESLA in our proposed scheme. TESLA is based on one-way hash chain (OHC) [19]. An OHC is a chain of keys generated through repeatedly applying a one-way hash function H on a random number k_Q as follows,

$$k_i = H(k_{i+1}) \; for \; 0 \leq i \leq Q-1$$

At the beginning k_0 is securely distributed to all receivers. Then, the source node releases the keys in its OHC chain in the reverse order of their generation, and use the keys to authenticate packets. For example, the i-th packet pkt_i is

$$pkt_i = \{P||H_{k_i}(P)||k_{i-1}\}$$

A packet cannot be verified until the following packet is received. For example, when a node receives pkt_i, it uses pkt_i to verify pkt_{i-1} and then buffers pkt_i. When it receives pkt_{i+1}, it first checks whether k_i in pkt_{i+1} satisfies $k_{i-1} = H(k_i)$. If yes, k_i is a good key. Then, it uses k_i to verify $\{P||H(P)\}_{k_i}$. If yes, pkt_i passes the verification. Because the key chain is released in the reverse order of the generation, it is computationally infeasible for any node other than the source to derive the keys in the chain that have not been released yet. This property makes attackers unable to forge the OHC. More details on applying OHC in authentication can be found in [17], [20].



(a) Dominating nodes (b) Trusted nodes

Fig. 1.

TESLA was extended to provide immediate source authentication by calculating authentication information prior to sending packets [17]. Furthermore, TESLA requires a digital signature operation to bootstrap itself, and thus is impractical in resource constrained sensor networks. Hence, μTESLA was developed to use symmetric cryptography to distribute initial parameters to the sensor nodes individually [21]. Multi-level μTESLA [22] was later proposed to reduce the high communication overhead required for initializing sensor nodes when the number of sensor nodes is large. All these works mainly focus on the bootstrap of TESLA in different applications. Our work focuses on reducing verification delay of TESLA, and is complementary to these works. In this paper, we will use any of these TESLA mechanisms to bootstrap our L-TESLA and LD-TESLA.

B. Dominating Set Protocols

A sensor network can be modeled as a graph $G = (V, E)$, where V is the set of vertexes (sensor nodes) and E is the set of edges (communication links). A *dominating set* for the graph G is a subset V' of V such that any node not in V' is directly linked to at least one node in V'. Nodes in V' are named *dominating nodes*. Furthermore, a *connected dominating set* of the graph G is a dominating set V_c' such that all nodes (including dominating nodes) are neighboring to at least one node in V_c'.

If a connected dominating set V_c' is found, one can form a spanning tree of G where V_c' forms the set of non-leaf nodes of the tree. Thereby, all nodes, including dominating nodes, are neighbors to at least one node in V_c'. A simple example of a connected dominating set is depicted in Figure 1(a), where nodes B, C and D are the dominating nodes. When broadcasting, only dominating nodes rebroadcast packets. Since all nodes are neighboring to dominating nodes, all nodes will receive the broadcast packet and broadcast overhead can be reduced. Hence, using dominating nodes in broadcast can both ensure network-wide delivery of messages and avoid unnecessary rebroadcasts from non-dominating nodes. Furthermore, if a minimum connected dominating set can be found, the broadcast overhead can be minimized. Hence, dominating sets have been critical in designing efficient broadcast and networking protocols [23], [24].

Finding the minimum connected dominating set in a given graph is generally a NP-complete problem, and solving the minimum dominating set problem has been an active

research area for many years [25]. Johnson [26] shows that in polynomial time the dominating set problem can be solved approximately within $1 + log|V|$ since the problem is a special instance of minimum set cover, but the problem is not approximable within $clog|V|$, for some $c > 0$ [27]. If the dominating set is restricted to be connected the problem is approximable within $logd+3$ where d is the maximum degree, and within $3log|V|$ for the vertex weighted version [28].

Dominating sets are useful in a number of applications that require efficient wireless broadcast operations to reduce the number of messages in the network. They are also applied in energy efficient protocols in sensor networks[23]. Another use of dominating sets is for routing in mobile ad-hoc networks. Nevertheless, finding the minimum connected dominating set in a given graph is in general an NP-complete problem, and solving the minimum dominating set problem has been an active research area for many years [25]. LD-TESLA is based on a distributed dominating set protocol [9] that finds local dominating set for each node in a network.

III. Preliminary

A. Verification Delay

Secure dissemination of broadcast data is an important aspect of applications in sensor networks. In TESLA, a receiving node needs to buffer packets, until the source discloses the corresponding key. The packets can be verified and accepted only after the receiving node receives the disclosed key. We consider two types of delays in this procedure. First is *verification delay* of a receiving node, i.e. the duration from when the node receives a packet to when the node receives the key for verifying the packet. Second is *application delay* of a receiving node, i.e. the duration from when the source broadcasts a packet to when the node receives the verification key. In TESLA, the verification delay is normally determined by the maximum transmission time for a broadcast packet to reach all nodes in a network to satisfy the security condition [1]. The verification delay determines how long a receiving node needs to buffer the packet. For a large verification delay, not only a received packet would be buffered for a longer time, but also more packets would be buffered in sensors for later verification. In a large network, the verification delay may be prohibitive, because it can cause storage problems in sensor nodes that need to buffer received packets. Reducing verification delay is thus a crucial issue to improve the performance of broadcast authentication.

B. Trusted Nodes

We notice that the large verification delay problem is mainly due to the assumption that only the broadcast source is secured and trusted. Whereas, in a real sensor network, sensors have different functions and capabilities and act as different roles. In many applications, we have a number of nodes that are better secured and trustable. For example, in the context of a battlefield sensor network, we can have a distributed three-tier wireless network structure [29]. The Sensor Tier consists of sensor nodes that may contain a number of autonomous sensors which can sense different types of data such as light, sound, movement, and image. Sensor nodes may be ground based or airborne. They are often non-trusted nodes. The Soldier Tier consists of soldier nodes (PDAs) carried by soldiers. Soldier nodes have more computing and communication capabilities than sensor nodes, but nevertheless still non-trusted since they are evolving in hostile environment. Soldier nodes may act as both sensor data sources (sensors hosted in the soldier node platform) and sinks (nodes that process data collected from variety of sensor nodes). The Backbone Tier consists of backbone nodes mounted on vehicles or weapon platforms. Backbone nodes have a lot more computing and communications capabilities than soldier nodes, and act as sensor data source, sinks and information relays. The backbone nodes are enough secured and can be considered as trusted nodes. In such a distributed multi-tier wireless sensor network environment, the trusted nodes can be used to help reduce the verification delay in broadcast authentication, since they are hard to compromise and have enough resources to authenticate packets on behalf of the real source.

C. Problem Statements

The problem we will address in this paper is
Given a sensor network with T trusted nodes and $V - T$ non-trusted nodes, how to reduce the verification delay and the broadcast overhead when broadcasting an authenticated packet?

To solve the problem, we propose to localize broadcast authentication by

1) asking only trusted nodes to re-authenticate broadcast packets in the name of the original source (in both L-TESLA and LD-TESLA)
2) asking only dominating nodes to rebroadcast (in LD-TESLA)
3) asking sensor nodes to verify packets relayed from their nearby trusted nodes (in both L-TESLA and LD-TESLA)

The key requirement in our approach is that only trusted nodes need to have computation and storage resource for the localized authentication while all normal sensor nodes should be able to verify packets as in TESLA without asking additional resource. The challenges of this approach lies in two facts. First, how sensor nodes can be self-organized into subsets without knowing the topology of the whole network. It is expected that each subset has one trusted node that will re-authenticate broadcast packets in the subset and the subset should be as small as possible. Second, how dominating sets can be adjusted according to the broadcast subsets so that trusted nodes are included into the dominating sets without significantly affecting the verification delay in the network.

D. Key Setup

In this paper, we ask each sensor node to set up two types of keys before broadcasting can be localized. First, each sensor node i establishes a secret key $S_{i,j}$ with each trusted node j for exchanging some messages. Since the number of trusted nodes is small, a simple approach is to preload the secret

keys into sensor nodes. Researchers also proposed several sophisticated approaches [30], [31], [32], [33], [34], [35]: a) Sensor nodes can remember the IDs of all trusted nodes and then establish keys with the trusted nodes based on their IDs; b) Sensor nodes can also be preloaded with a number of keys randomly selected from a large key pool, and then establish keys based on common preloaded keys. Second, each sensor node i establishes a secret key $s_{i,j}$ with each of its one-hop neighboring nodes. As proposed in [32], [36], [35], a sensor node can setup keys with its one-hop neighboring nodes with a one-time secret. The keys are used for exchanging DOM messages among neighboring nodes.

Finally, since trusted nodes have more computation, storage and power resources, we let each trusted node generate and maintain two key chains: a global chain and a local chain. The global chain is applied when constructing a secure broadcast mesh for normal nodes to find their nearby trusted nodes. The local chain is then applied for trusted nodes to locally authenticate broadcast packets and for normal nodes to locally verify them.

E. Notations and Key Setup

To present the detailed protocols, we use the following notations.

- $k_{i,t}$: the t-th key in the local key chain generated by trusted node i, and $k_{i,t-1} = H(k_{i,t})$.
- $K_{i,t}$: the t-th key in the global key chain generated by trusted node i, and $K_{i,t-1} = H(K_{i,t})$.
- $S_{i,j}$: a secret key shared between node i and trusted node j.
- $s_{i,j}$: a secret key shared between node i and its neighboring node j.
- m: a message.
- $H_k(m)$: a hash of message m with key k.
- $E_k(m)$: an encryption of message m with key k.
- D: upper bound of the delay in the network.
- d_j: upper bound of the delay in trusted node j's local area.
- q: the sequence number of a packet.

IV. SCHEME I: L-TESLA

A. Overview of L-TESLA

L-TESLA is designed to address the problem: *from which trusted node a node should receive local TESLA keys.* In Figure 1(b), N1 has three alternative trusted nodes to receive a broadcast packet from S: T1, T2 and T3. Although T3 is closest to N1, T3 is behind N1, i.e. T3 receiving packets from S later than N1. Hence, N1 should choose one from T1 and T2. T2 will be selected because T2 is closer to N1 than T1. Similar to N1, T3 also needs to choose a trusted node and cannot simply consider the closest trusted node. T3 will choose T2, because T2 receives broadcast packets earlier than T3 and T2 is closer to T3 than T1.

Problem of L-TESLA can be illustrated in Figure 1(b) as well. N1 normally receives packets from T1 first, and will take T1 as the trusted node. Later, packets rebroadcast by T2 will reach N1. Since T2 is a closer trusted node, N1 will choose T2 and rebroadcast T2's packets to N2 and T3 as they should also be aware that T2 is a closer trusted nodes. This update procedure doubles or triples broadcast overhead (depending on how many updates). The problem will be addressed in the proposed LD-TESLA. Note that LD-TESLA cannot completely replace L-TELSA, since they have different tradeoff between delay and overhead.

L-TESLA works in two phases. In the first phase, each trusted node broadcasts a request ($BREQ$) which is secured by its global key chain. The $BREQ$ helps all other nodes to find their nearby trusted nodes for this trusted source. Thereby, the area is divided into several local areas. In each local area, a trusted node can use its L-TESLA to rebroadcast messages while reducing the verification delay in the second phase. After all trusted nodes broadcast $BREQ$, a secure broadcast mesh is established in the area so that:

(a) Each trusted node has a verification delay and a local key chain for its local TESLA, and
(b) All nodes know from which upstream trusted node they should receive and verify a broadcast message according to the source of the message.

In another words, the first phase uses the traditional TESLA to bootstrap local TESLA in the second phase.

In the second phase, when a node needs to broadcast a message ($BPKT$), it first sends the message to its closest trusted node. Then, the trusted node broadcasts the message authenticated by its local key chain. Upon receiving the broadcast message, a normal node will rebroadcast the message as long as the message is secure, first time received, and from its selected upstream trusted node. Nevertheless, a trusted node will transform the message with its own local TESLA information. Note that all nodes verify the $BPKT$ based on the local TESLA of their selected upstream trusted node.

B. First Phase of L-TESLA

After deployment, each trusted node i broadcasts a request ($BREQ$) which is secured by TESLA with its global key chain. K_{i,t_i} is a key in trusted node i's global key chain. The interval to disclose the verification key K_{i,t_i} is the upper bound of the delay in the whole network, which is known by all nodes in the network.

$$\{[i, t_i, H_{K_{i,t_i}}(i, t_i), K_{i,t_i-1}]\}$$

When the $BREQ$ is rebroadcast by a trusted node j, the packet will have two segments as follows. The first segment is the initial $BREQ$ indicating which trusted node initiates the request. The second segment indicates that trusted node j rebroadcasts the request and p_j is j's previous trusted node. Both segments are authenticated by the corresponding trusted nodes as in TESLA.

$$\{[i, t_i, H_{K_{i,t_i}}(i, t_i), K_{i,t_i-1}][j, p_j, t_j, H_{K_{j,t_j}}(j, p_j, t_j), K_{j,t_j-1}]\}$$

Upon receiving the $BREQ$ at time t_r, node k processes it according to Algorithm 1. First, it checks whether K_{i,t_i} and K_{j,t_j} have been disclosed. If yes, the $BREQ$ might not be secure and must be discarded. Node k processes the $BREQ$ only if it is secure. Then, if the $BREQ$ is a new request (i.e. the first segment in $BREQ$ was not received before), node k

Algorithm 1 Process a $BREQ$ in L-TESLA

Assuming node k receives a $BREQ$ at t_r.
$\{[i, t_i, H_{K_{i,t_i}}(i, t_i), K_{i,t_i-1}][j, p_j, t_j, H_{K_{j,t_j}}(j, p_j, t_j), K_{j,t_j-1}]\}$
Node k will do the following.
if $t_i + D < t_r$ **then**
  Discard the $BREQ$ and return
end if
if This is the first time receiving the $BREQ$ **then**
  Create an entry R_i for this $BREQ$
  Record i, j, q, $e_t = t_r - t_j$ and $e_s = t_r - t_i$ in R_i
  if Node k is a trusted node **then**
    Update the $BREQ$ to
      $\{[i, t_i, H_{K_{i,t_i}}(i, t_i), K_{i,t_i-1}][k, j, t_k, H_{K_{k,t_k}}(k, j, t_k), K_{k,t_k-1}]\}$
  end if
else
  Find R_i corresponding to $BREQ$'s i and q
  if $t_r - t_j < e_t$ and $t_j - t_i < e_s$ and R_i's j is p_j **then**
    Update j, e_t and e_s in R_i
  else
    Discard the $BREQ$ and return
  end if
end if
Rebroadcast the $BREQ$

Algorithm 2 Process a $BPKT$ in L-TESLA

Assuming node k receives a $BPKT$ at t_r
$\{[i, j, t'_j, m, H_{k_{j,t'_j}}(i, j, m), k_{j,t'_j-1}]\}$
Node k will do the following.
find R_i corresponding to $BPKT$'s i and q
if R_i and the $BPKT$ have different j **then**
  Discard the $BPKT$
end if
if The $BPKT$ was received or $t_r - t'_j > d_j$ **then**
  Discard the $BPKT$ and return
end if
if Node k is a trusted node **then**
  Wait for the verification key disclosed by j
  if The $BPKT$ does not pass verification **then**
    Discard the $BPKT$ and return
  end if
  Update the $BPKT$ to
    $\{[i, k, t'_k, m, H_{k_{k,t'_k}}(i, k, m), k_{k,t'_k-1}]\}$
end if
Rebroadcast the $BREQ$

creates an entry R_i to record the source ID i, the upstream trusted node ID j, the delay from j to k, and the delay from i to k. If the $BREQ$ was previously received (from another upstream trusted node), the node will accept the $BREQ$ to update the corresponding R_i if (1) j is closer to k than the upstream trusted node in R_i and (2) the upstream trusted node in R_i is also j's upstream trusted node p_j, i.e. j is the only trusted node between k and p_j.

The $BREQ$ will be rebroadcast only in one of the two conditions: (1) it is a new request; or (2) it carries a closer upstream trusted node. If a trusted node rebroadcasts the $BREQ$, it will replace the second segment of the $BREQ$ with its own information. Hence, whenever a node receives the $BREQ$, it always knows which trusted node is in the upstream. Note that the first segment of the $BREQ$ will not be changed in the whole procedure.

After the $BREQ$ reaches all nodes and all trusted nodes disclose the keys in the $BREQ$, all nodes verify the received $BREQ$s. If the $BREQ$s are secure, node k chooses its closest upstream trusted node j for the source trusted node i. Node k sends the delay e_t in E_i to node j via a reply message $BREP$, $\{E_{S_{k,j}}(j, e_t, H(j, e_t))\}$, which is secured by the secret key $S_{k,j}$. Then, each trusted node j takes the maximum of the delays in $BREP$s from its nearby nodes as the verification delay d_j in its local TESLA. Each trusted node broadcasts the verification delay to its nearby nodes and bootstraps its local TESLA.

C. Second Phase of L-TESLA

The second phase is for a source node to broadcast a message after the secure broadcast mesh has been constructed in the first phase. For simplicity, we only consider a trusted node as a source node. If a normal node wants to broadcast a message, it first sends the message to its nearby trusted node. Then, the trusted node broadcasts the message. This can simplify the function of normal nodes.

Assuming a trusted node j rebroadcast a packet $BPKT$ whose source is the trusted node i. The $BPKT$ contains the following information.

$$\{[m, i, j, H_{k_{j,t'_j}}(m, i, j), k_{j,t'_j-1}]\}$$

Node j actually uses its local TESLA to authenticate the $BPKT$.

Upon receiving the $BPKT$, node k processes it according to Algorithm 2. It first checks whether node j is the upstream trusted node from which the packet should come for the source i. If no, the packet should be discarded to avoid redundant rebroadcasting. Node k also checks whether the verification key was disclosed by node j. If yes, the $BPKT$ is not secure and should be discarded. After passing these two checks, the $BPKT$ will be rebroadcast if it is the first time to receive the packet. A normal node will rebroadcast the $BPKT$ before the verification key is disclosed. However, a trusted node will wait for the verification key to verify the $BPKT$ before rebroadcasting it. While rebroadcasting, a trusted node will use its own local TESLA to authenticate the $BPKT$, and thus recompute and replace the authentication information in the $BPKT$.

D. Example

An example of L-TESLA is depicted in Figure 2, where A and B are trusted nodes and the others are normal nodes. The two subfigures illustrate the local areas for A and B to use their local TESLA. In Figure 2(a), the gray nodes should be covered by A's local TESLA. For example, A should ensure that a $BPKT$'s key cannot be disclosed until the $BPKT$

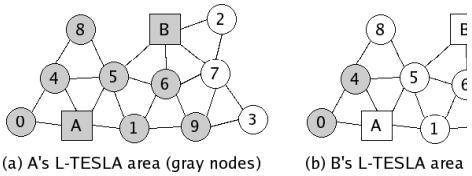


Fig. 2. Example of L-TESLA

reaches nodes B, 6 and 9. Similarly in Figure 2(b), the white nodes should be covered by B's local TESLA.

The figure shows that the nodes in the area between A and B are covered by both trusted nodes. However, these nodes do not rebroadcast $BPKT$s more than once. Instead, they rebroadcast according to the packet source and the upstream trusted node. For example, if A is the broadcast source, node 5 will only rebroadcast the $BPKT$ received from A. When B receives and rebroadcasts the $BPKT$, node 5 will not rebroadcast the $BPKT$ again, even though node 5 is covered by B's L-TESLA.

Note that, in the first phase of L-TESLA, when A broadcasts a $BREQ$, node 3 will first receive the $BREQ$ from node 9. At this time, node 3 will perceive that A is its closest upstream trusted node. However, node B will rebroadcast the $BREQ$ to node 7 and then to node 3 again. Hence, in a later time, node 3 will update its closest upstream trusted node to B.

V. Scheme II: LD-TESLA

LD-TESLA is designed to address the problem: *which node should function as a dominating node in broadcast authentication?* It integrates dominating set protocols into L-TESLA to include two features: (1) trusted nodes must act as dominating nodes since they always need to rebroadcast in LD-TESLA; (2) nodes are adjusted regarding whether or not they are dominating in specific circumstances. The differences of LD-TESLA and L-TESLA are (a) LD-TESLA requires sensor nodes to find local dominating nodes and (b) LD-TESLA asks only dominating nodes to rebroadcast.

A. Localized Dominating Sets

When being deployed, sensor nodes can use any dominating set protocol [8] to identify the dominating set of the network. This paper develops a distributed protocol based on [9]: every node individually selects a dominating set from its one-hop neighboring nodes so that any two-hop neighboring node is a neighbor of at least one of the dominating nodes. When nodes are deployed in the network, they first obtain the IDs of their one-hop and two-hop neighboring nodes. As discussed in [32], [36], the security of this procedure together with establishing secret keys $s_{i,j}$ can be ensured by one-time secret preloaded in sensor nodes. Then, each node u executes Algorithm 3 to find its own dominating set. Starting with $D(u) = \emptyset$, the algorithm adds trusted nodes into its dominating set first, and then selects other dominating nodes from its one-hop normal neighboring nodes to $D(u)$ until all two-hop neighboring nodes are covered by $D(u)$. Each node finally notifies its neighbors of its selection and the notification messages are secured by $s_{i,j}$. The protocol ensures that trusted nodes will

Algorithm 3 Find node u's dominating set in LD-TESLA

$D(u) = \emptyset$
$V(u)$=$\{$all one-hop neighboring nodes of node $u\}$
$W(u)$=$\{$all two-hop neighboring nodes of node $u\}$
Find $T(u) \subseteq V(u)$ such that $T(u)$ includes all trusted nodes that are one-hop neighboring to u.
Find $Q(u) \subseteq W(u)$ such that $Q(u)$ includes all u's two-hop neighboring nodes that are one-hop neighboring to nodes in $T(u)$.
$D(u) = D(u) + T(u)$
$V(u) = V(u) - T(u)$
$W(u) = W(u) - Q(u)$
while $W(u)$ is not empty **do**
  Find $v \in V(u)$ such that $|N(v)|$ is maximized where $N(v)$ is the set of v's one-hop neighboring nodes and $N(v) \subseteq W(u)$.
  $D(u) = D(u) + \{v\}$
  $V(u) = V(u) - \{v\}$
  $W(u) = W(u) - N(v)$
end while
Notify nodes in $D(u)$ to function as dominating nodes for node u.

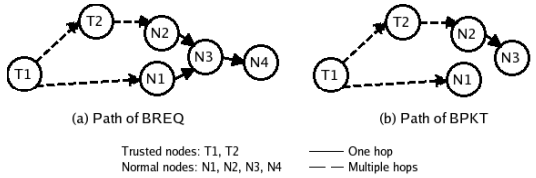


Fig. 3. Illustration of problems caused by dominating nodes

work as dominating nodes and all nodes in the network can receive broadcast packets with significantly reduced broadcast overhead.

The algorithm can be easily implemented in sensor nodes. However, since each node individually and locally selects the dominating nodes from its neighboring nodes, a node could be a dominating node for some neighboring nodes but not for the others. When applying L-TESLA with this dominating set protocol, a network could be disconnected regarding broadcast. As shown in Figure 3(a), $N3$ is $N1$'s dominating node, but not $N2$'s dominating node. Assume a trusted node $T1$ broadcasts a request packet $BREQ$ in the network to establish a broadcast mesh (more details could be found in [37]). At t_1, $N3$ receives the $BREQ$ rebroadcast by $N1$. Because $N3$ is $N1$'s dominating node, $N3$ will rebroadcast the packet to $N4$. So, both $N3$ and $N4$ will select $T1$ as their nearby trusted node. However, later on at t_2, $N3$ receives the $BREQ$ rebroadcast from another path via $T2$ and $N2$. Because the later received $BREQ$ has a shorter verification delay as $T2$ is closer to $N3$ than $T1$, $N3$ will select $T2$ as its nearby trusted node. However, $N3$ will not rebroadcast the $BREQ$ as it is not $N2$'s dominating node. Consequently, $N4$ is still keeping $T1$ as its nearby trusted node. After such a broadcast mesh is established, if $T1$ broadcasts a data packet $BPKT$, $N3$ will neither accept the $BPKT$ received from $N1$ as $N3$ and $N1$ have different nearby trusted nodes, nor rebroadcast the $BPKT$ received from $N2$ as $N3$ is not $N2$'s dominating

Algorithm 4 Process a $BREQ$ in LD-TESLA

Assuming node k receives a $BREQ$ at t_r.
$\{[i, t_i, H_{K_{i,t_i}}(i, t_i), K_{i,t_i-1}][j, p_j, t_j, H_{K_{j,t_j}}(j, p_j, t_j), K_{j,t_j-1}]\}$
Node k will do the following.

if (C1): $t_i + D < t_r$ **then**
  Discard the $BREQ$ and return
end if
Let h be the previous hop to k
if (C2): This is the first time receiving the $BREQ$ **then**
  Create an entry R_i for this $BREQ$
  Record i, j, h, q, $e_t = t_r - t_j$ and $e_s = t_r - t_i$ in R_i
else
  Find R_i corresponding to $BREQ$'s i and q
  if (C3): $t_r - t_j < e_t$ and $t_j - t_i < e_s$ and R_i's j is p_j **then**
    if Node k is a dominating node for R_i's h, but not for the current h **then**
      Promote k as a dominating node for current h
    end if
    Update j, h, e_t and e_s in R_i
  else
    Discard the $BREQ$ and return
  end if
end if
if Node k is a trusted node **then**
  Update the $BREQ$ to
    $\{[i, t_i, H_{K_{i,t_i}}(i, t_i), K_{i,t_i-1}][k, j, t_k, H_{K_{k,t_k}}(k, j, t_k), K_{k,t_k-1}]\}$
end if
if Node k is a trusted node or a dominating node **then**
  Rebroadcast the $BREQ$
end if

node. Hence, as depicted in Figure 3(b), the $BPKT$ broadcast by $T1$ cannot reach $N4$, even though $N3$ receives two copies of the $BPKT$.

B. First Phase of LD-TESLA

To address the problem caused by dominating nodes, we propose Algorithm 4 based on L-TESLA's first phase algorithm to adjust the dominating status of nodes. To localize broadcast authentication with dominating nodes, each trusted node i still needs to broadcast a request $BREQ$. Upon receiving the $BREQ$ at time t_r, node k processes it with three checks: (C1) whether the $BREQ$ is secure, (C2) whether the $BREQ$ is new, and (C3) whether the $BREQ$ carries information of a closer trusted node. The $BREQ$ can only be rebroadcast by dominating nodes (including all trusted nodes) if it is secure and new or carrying a closer trusted node.

Nevertheless, a normal node k may need to adjust its dominating status in the third check if the $BREQ$ does carry information of a closer trusted node. Recall that node k keeps a record for its nearby trusted node. If it finds a closer trusted node, it needs to update the record. The record also shows whether it is a dominating node for the previous hop from which it receives the $BREQ$. As illustrated in Figure 3, it is possible that node k was a dominating node in the old record, but not in the updated record. In such a situation, node k promotes itself as a dominating node in the updated record and demotes in the old record. Obviously, in other cases, node k does not need to adjust. The consequence of the adjustment has two folds. First, nodes in the downstream of node k can update their nearby trusted nodes according to the $BREQ$. Second, future data packets $BPKT$s can still reach nodes in the next hop of node k, as node k is still a dominating node.

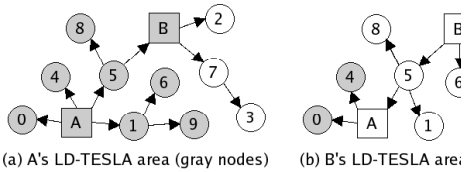


Fig. 4. Example of LD-TESLA

After the $BREQ$ reaches all nodes and all trusted nodes disclose the keys in the $BREQ$, all nodes will verify the received $BREQ$. If the $BREQ$ is secure, node k chooses its closest upstream trusted node j for the source trusted node i. Node k sends the delay e_t in R_i to node j via a reply message $BREP$, $\{E_{S_{k,j}}(j, e_t, H(j, e_t))\}$, which is secured by the secret key $S_{k,j}$. Then, each trusted node j takes the maximum of the delays in the received $BREP$s from its nearby nodes as the verification delay d_j for its local TESLA keys. Each trusted node broadcasts the verification delay to its nearby nodes and bootstraps its local TESLA via typical procedures [2], [22].

The area is thus divided into several local areas, and a secure broadcast mesh is established in the area so that:

(a) Each trusted node has a verification delay and maintains a local key chain for its local TESLA.
(b) All nodes know from which upstream trusted node they should receive and verify a broadcast packet according to the source of the packet.
(c) Only dominating nodes present in the secure broadcast mesh and rebroadcast packets.

The second phase of LD-TESLA (i.e., authentication and broadcast of data packet $BPKT$s) would be the same as L-TESLA's second phase according to Algorithm 2.

C. Example

Figure 4 shows a scenario similar to Figure 2, but LD-TESLA is applied in this scenario to localize broadcast authentication. In the figure, A and B are trusted nodes and the others are normal nodes. The two subfigures illustrate the local areas for A and B to use their local TESLA keys. In Figure 4(a), the gray nodes should be covered by A's local TESLA keys. For example, A should ensure that a $BPKT$'s key cannot be disclosed until the $BPKT$ reaches nodes B, 6 and 9. Similarly in Figure 4(b), the white nodes should be covered by B's local TESLA.

The figure shows that the nodes in the area between A and B are covered by both trusted nodes. However, these nodes do not rebroadcast $BPKT$s more than once. Instead, they rebroadcast according to the packet source and the upstream trusted node. For example, if A is the broadcast source, node 5 will only rebroadcast the $BPKT$ received from A. When

B receives and rebroadcasts the $BPKT$, node 5 will not rebroadcast the $BPKT$ again, even though node 5 is in B's local area. In addition, only nodes 5 and 1 will rebroadcast packets received from A, because they are A's dominating nodes.

Note that, in the first phase of LD-TESLA, when A broadcasts a $BREQ$, node 3 will first receive the $BREQ$ from node 9. At this time, node 3 will perceive that A is its closest upstream trusted node. However, node B will rebroadcast the $BREQ$ to node 7 and then to node 3 again. Hence, in a later time, node 3 will update its closest upstream trusted node to B. Finally, all nodes in Figure 2 rebroadcast, but only dominating nodes in Figure 4 rebroadcast.

VI. Evaluation

In this section, we first analyze the security of the proposed protocols and discuss their limitations. Then, we use simulation to evaluate their performance and discuss their applicability.

A. Security Analysis

In security analysis, we do not assume that attackers are able to compromise keys used in bootstrapping L-TESLA and LD-TESLA and authenticating broadcast packing with local TESLA key chains, since the corresponding cryptographic algorithms are hard to break. Thereby, we consider a few scenarios attackers may exploit to compromise L-TESLA and LD-TESLA. We mainly study whether attackers can forge broadcast packets and how attacks can affect verification delay and broadcast overhead in the analysis.

a) Scenario 1: Trusted nodes are not trustable.: This is the worst scenario, since the proposed two protocols heavily rely on the security of trusted nodes. A trusted node may be compromised and then broadcast false information. Since L-TESLA and LD-TESLA need trusted nodes to re-authenticate packets, a compromised trusted node can modify or forge any packet. Thereby, trusted nodes should be sufficiently protected if the proposed schemes will be deployed. As discussed in Section III-B, certain nodes in a sensor network have the capability to function as trusted nodes not only because they have abundant computing capability but also because they are normally much better secured than normal sensor nodes. Hence, in the following scenarios, we will focus the security analysis on the situations where only normal nodes may be compromised.

b) Scenario 2: Dominating nodes are not dominating.: In LD-TESLA, any sensor node may be a dominating node. A compromised node may claim itself as a dominating node in either Algorithm 3 where nodes are identifying nearby dominating nodes or Algorithm 4 where a node may promote itself as a dominating node. Apparently, false dominating nodes cannot forge valid broadcast packets, since only trusted nodes have credentials for authentication. False dominating nodes neither can deceive other nodes for verification delay, since delay information is authenticated by trusted nodes. Hence, false dominating nodes can mostly change the broadcast topology but not the verification delay. False dominating nodes may increase broadcast overhead a little as the nodes that receive packets from false dominating nodes may rebroadcast. Nevertheless, only rebroadcast from nodes one-hop to false dominating nodes adds additional overhead. Nodes multi-hop away from false dominating nodes do not make extra overhead.

	LD-T	L-T	DOM [9]	TESLA [2]
Authentication	Y	Y	N	Y
Localization	Y	Y	N	N
Dominating	Y	N	Y	N

TABLE I
COMPARISON OF FOUR PROTOCOLS

B. Performance Analysis

In this section, we study the performance of L-TESLA and LD-TESLA by simulation. The simulation setting is described first, and then results for different performance metrics are presented for L-TESLA and LD-TESLA.

1) Setting: We implemented a discrete event simulator for verifying and evaluating the proposed L-TESLA and LD-TESLA protocols. Four protocols are implemented and compared: L-TESLA, LD-TESLA, TESLA [2] and DOM [9]. Their major features and differences are summarized in Table I.

In the simulation, we set the number of all nodes $N = 1000$ (i.e. deploy 1000 nodes) in a $5000 \times 5000m^2$ area. Each node has a transmission range of $250m$, and a random delay in the range of $0s$ to $0.5s$. We vary the number of trusted nodes T from 4 to 49 in order to study the improved performance of L-TESLA and LD-TESLA. We average measurement over 30 randomly generated experiments.

Nodes are deployed via two representative approaches. One is random deployment, i.e. all nodes (trusted nodes and normal nodes) are randomly deployed in the area. The position of the nodes follows the uniform deployment. The other is even deployment, i.e. the area is evenly divided into 1000 grids. Each node is deployed at one grid. Then, the whole area is evenly divided into T sub areas so that each trusted node is deployed in the center of one sub area.

Three metrics are measured to evaluate the performance of L-TESLA and LD-TESLA: verification delay, broadcast overhead of $BREQ$, broadcast overhead of $BPKT$. Verification delay is defined in Section III-A. Both L-TESLA and LD-TESLA target reducing the verification delay so that a node can save the resource for buffering unverified packets. Broadcast overhead of $BREQ$ is measured as the number of broadcast requests for constructing a broadcast mesh. This metric shows the initial broadcast overhead for the whole network to set up L-TESLA or LD-TESLA for later broadcast. Broadcast overhead of $BPKT$ is measured as the number of broadcast data packets. Compared with L-TESLA, LD-TESLA is expected to the broadcast overheads simultaneously to save sensor's resources, but may not reduce the verification delay as much as L-TESLA.

2) Verification delay: Figure 5 shows that verification delay is smaller in even deployment than in random deployment for all protocols. This is mainly because even deployment better covers the area than random deployment. In a random

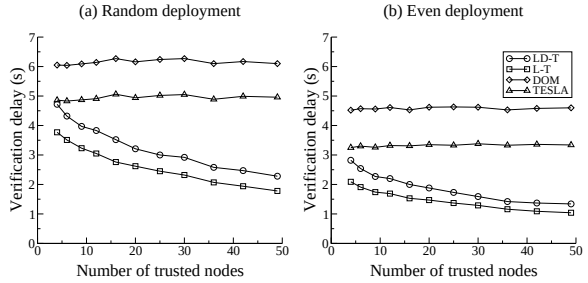


Fig. 5. Comparison of verification delay

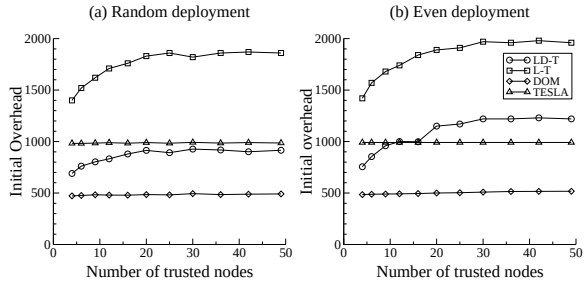


Fig. 6. Comparison of broadcast overhead of $BREQ$

deployment, it is quite possible that several trusted nodes are closely deployed and thus some areas have no close trusted nodes to cover. Nodes in these areas have to receive packets and wait for verification keys from distant trusted nodes.

The figure also illustrates that LD-TESLA and L-TESLA well reduces the verification delay, because both of them localize broadcast authentication by using local TESLA keys of trusted nodes and normal nodes only need to wait for the verification key sent from their nearby trusted nodes. As the number of trusted nodes increases, the verification delay decreases. Because the verification delay in L-TESLA should be approximately proportional to the distance to the nearby trusted node, the delay is inverse to the square root of the number of trusted nodes. In contrast, verification delays of TESLA and DOM keep flat in both deployments.

3) Broadcast Overhead of $BREQ$: Figures 6 illustrate the broadcast overheads of $BREQ$. Without using dominating nodes, L-TESLA can almost double the broadcast overhead of $BREQ$, which indicates that many nodes will update and rebroadcast when they find closer trusted nodes. By following each $BREQ$, we found that a) only nodes around source will not rebroadcast, and b) $BREQ$s might be rebroadcast over several hops in network to update downstream nodes on closer trusted nodes. Without updating, the downstream nodes may still use trusted nodes that are actually further away. Hence, localizing broadcast authentication needs more broadcast overhead of $BREQ$ due to the update broadcast. LD-TESLA matches TESLA in terms of broadcast overhead of $BREQ$.

Note that using dominating nodes does not eliminate the update broadcast. In stead, dominating nodes save the overhead of rebroadcast for the update. Because only dominating nodes can broadcast, $BREQ$s are rebroadcast by fewer nodes. At the same time, all downstream nodes can receive the updated information as dominating nodes should provide a full coverage of the network.

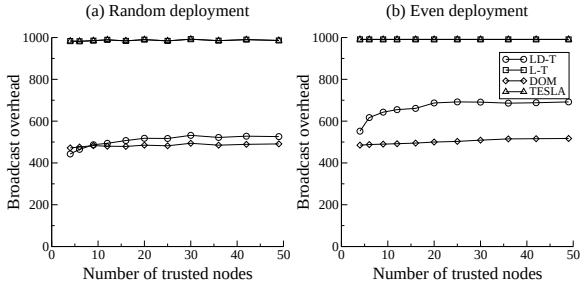


Fig. 7. Comparison of broadcast overhead of $BPKT$

4) Broadcast Overhead of $BPKT$: Figures 7 illustrate the broadcast overheads of $BPKT$. Obviously, DOM can reduce almost half of the broadcast overheads. By integrating it into localized broadcast authentication, LD-TESLA well outperforms L-TESLA and TESLA in terms of broadcast overhead of $BPKT$. When broadcasting normal packets $BPKT$, LD-TESLA's overhead is close to DOM, because only dominating nodes need to rebroadcast. Furthermore, it is noticed that LD-TESLA performs better in Figure 7(a) than in Figure 7(b). In even deployment, trusted nodes are better distributed, and nodes have more alternative paths for smaller verification delay. Consequently, more non-dominating nodes may be adjusted as dominating nodes in LD-TESLA, when closer trusted nodes are found. Thereby, the broadcast overhead of $BPKT$ will be higher in even deployment than in random deployment.

In summary, both L-TESLA and LD-TESLA can significantly reduce verification delay compared with TESLA. However, L-TESLA demands more broadcast overhead in sensor networks, although it achieves the best reduction of verification delay. In contrast, LD-TESLA demands less broadcast overhead, while its delay reduction is not as good as L-TESLA. Therefore, the deployment of the two protocols is determined by the priority of applications. If small delay is more preferred, L-TESLA should be selected. Otherwise, if both delay and overhead should be considered, LD-TESLA should be the choice.

VII. Conclusion

This paper proposes L-TESLA and LD-TESLA for broadcast authentication in large multi-hop sensor networks. The L-TESLA algorithm extends TESLA by using a number of trusted nodes in the network. In a first step we construct a covering of the set of nodes in the network, where the number of subsets is equal to the number of trusted nodes. Then a local delay is computed for each subset and is used for broadcasting messages. Then we extended L-TESLA by taking advantage of techniques related to dominating sets. In this case, only dominating nodes rebroadcast packets. The simulations we performed show a significant reduction in verification delay of L-TESLA and LD-TESLA compared with non-localized broadcast authentication TESLA. L-TESLA and LD-TESLA also provide tradeoff between delay and overhead. L-TESLA reduces verification delay at the expense of increasing broadcast overhead; while LD-TESLA reduces broadcast overhead but cannot reduce delay as much as L-TESLA.

REFERENCES

[1] A. Perrig, R. Canetti, D. Tygar, and D. Song, "The tesla broadcast authentication protocol," *Cryptobytes*, vol. 5, no. 2, pp. 2–13, 2002.

[2] A. Perrig, D. Song, R. Canetti, J. D. Tygar, and B. Briscoe, "Timed efficient stream loss-tolerant authentication," IETF RFC4082, http://www.ietf.org/rfc/rfc4082.txt, 2005.

[3] C. Karlof, N. Sastry, Y. Li, A. Perrig, and D. Tygar, "Distillation codes and applications to dos resistant multicast authentication," in *NDSS*, 2004, pp. 37–56.

[4] A. Lysyanskaya, R. Tamassia, and N. Triandopoulos, "Multicast authentication in fully adversarial networks," in *IEEE Symposium on Security and Privacy*, 2004.

[5] J. M. Park, E. K. P. Chong, and H. J. Siegel, "Efficient multicast packet authentication using signature amortization," in *IEEE Symposium on Security and Privacy*, 2002, p. 227.

[6] D. Song, D. Zuckerman, and J. Tygar, "Expander graphs for digital stream authentication and robust overlay networks," in *IEEE Symposium on Security and Privacy*, 2002, pp. 241–253.

[7] C. K. Wong and S. S. Lam, "Digital signatures for flows and multicasts," *IEEE/ACM Transactions on Networking*, vol. 7, no. 4, pp. 502–513, 1999.

[8] S. Kutten and D. Peleg, "Fast distributed construction of k-dominating sets and applications," *Journal of Algorithms*, vol. 28, pp. 40–66, 1998.

[9] P.-J. Wan, K. M. Alzoubi, and O. Frieder, "Connected dominating set in wireless ad hoc networks," in *IEEE INFOCOM*, vol. 3, no. 1597-1604, June 2002.

[10] V. Mhatre and C. Rosenberg, "Homogeneous vs. heterogeneous clustered sensor networks: A comparative study," in *IEEE ICC*, 2004.

[11] N. Vlajic and D. Xia, "Wireless sensor networks: To cluster or not to cluster?" in *WOWMOM*, 2006, pp. 258–268.

[12] H. Chan, A. Perrig, and D. Song, "Secure hierarchical in-network aggregation in sensor networks," in *ACM CCS*, 2006, pp. 278–287.

[13] S. Zhao, I. Seskar, and D. Raychaudhuri, "Performance and scalability of self-organizing hierarchical ad hoc wireless networks," in *IEEE WCNC*, vol. 1, 2004, pp. 132–137.

[14] Y. Challal, A. Bouabdallah, and Y. Hinard, "Efficient multicast source authentication using layered hash-chaining scheme," in *IEEE International Conference on Local Computer Networks*, 2004, pp. 411–412.

[15] R. Gennaro and P. Rohatgi, "How to sign digital streams," in *Advances in Cryptology, CRYPTO*, 1997, pp. 180–197.

[16] J. M. Park, E. K. P. Chong, and H. J. Siegel, "Efficient multicast stream authentication using erasure codes," *ACM Transactions on Information and System Security*, vol. 6, no. 2, pp. 258–285, 2003.

[17] A. Perrig, R. Canetti, D. Song, and D. Tygar, "Efficient and secure source authentication for multicast," in *NDSS*, 2001, pp. 35–46.

[18] P. Golle and N. Modadugu, "Authenticating streamed data in the presence of random packet loss," in *NDSS*, 2001, pp. 13–22.

[19] L. Lamport, "Password authentication with insecure communication," *Communications of the ACM*, vol. 24, no. 11, pp. 770–772, 1981.

[20] A. Perrig, R. Canetti, J. Tygar, and D. Song, "Efficient authentication and signing of multicast streams over lossy channels," in *IEEE Symposium on Security and Privacy*, Berkeley, CA, 2000, pp. 56–73.

[21] A. Perrig, R. Szewczyk, V. Wen, D. Culler, and J. Tygar, "SPINS: security protocols for sensor networks," in *ACM MobiCom*, 2001, pp. 189–199.

[22] D. Liu and P. Ning, "Multi-level uTESLA: a broadcast authentication system for distributed sensor networks," *ACM Transactions on Embedded Computing Systems*, vol. 3, no. 4, pp. 800–836, 2004.

[23] J. Wu and H. Li, "On calculating connected dominating set for efficient routing in ad hoc wireless networks," in *International Workshop on Discrete Algorithms and Methods for Mobile Computing and Communications*, 1999, pp. 7–14.

[24] J. Wu and F. Dai, "A generic distributed broadcast scheme in ad hoc wireless networks," vol. 53, no. 10, 2004.

[25] S. Hedetniemi and R. Laskar, "Bibliography on domination in graphs and some basic definitions of domination parameters," *Discrete Math*, vol. 86, pp. 257–277, 1990.

[26] D. S. Johnson, "Approximation algorithms for combinatorial problems," *Journal of Computer System Science*, pp. 256–278, 1974.

[27] R. Raz and S. Safra, "A sub-constant error-probability low-degree test, and sub-constant error-probability pcp characterization of np," in *ACM Symposium on Theory of Computing*, 1997, pp. 475–484.

[28] S. Guha and S. Khuller, "Approximation algorithms for connected dominating sets," in *European Symposium on Algorithms, Lecture Notes in Computer Science*, vol. 1136, 1996, pp. 179–193.

[29] Q. Xue and A. Ganz, "Vertical communication in multimedia multi-tier tactical networks," in *IEEE MILCOM*, Oct. 2003.

[30] W. Du, J. Deng, Y. S. Han, and P. K. Varshney, "A pairwise key predistribution scheme for wireless sensor networks," in *ACM CCS*, 2003, pp. 42–51.

[31] L. Eschenauer and V. D. Gligor, "A key-management scheme for distributed sensor networks," in *ACM CCS*, 2002, pp. 41–47.

[32] S. Zhu, S. Setia, and S. Jajodia, "Leap: efficient security mechanisms for large-scale distributed sensor networks," in *ACM CCS*, Washington D.C., USA, 2003, pp. 62–72.

[33] D. Liu, P. Ning, and K. Sun, "Efficient self-healing group key distribution with revocation capability," in *ACM CCS*, 2003, pp. 231–240.

[34] D. Liu and P. Ning, "Establishing pairwise keys in distributed sensor networks," in *ACM CCS*, 2003, pp. 52–61.

[35] J. Deng, C. Hartung, R. Han, and S. Mishra, "A practical study of transitory master key establishment for wireless sensor networks," in *SECURECOMM*, 2005, pp. 289–302.

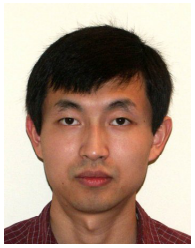
[36] H. Chan and A. Perrig, "Pike: peer intermediaries for key establishment in sensor networks," in *IEEE Infocom*, 2005.

[37] Jawad Drissi and Qijun Gu, "Localized broadcast authentication in large sensor networks," in *ICNS*, 2006.

[38] Y.-C. Hu, A. Perrig, and D. B. Johnson, "Ariadne: a secure on-demand routing protocol for ad hoc networks," in *ACM MobiCom*, 2002, pp. 12–23.

[39] ——, "Rushing attacks and defense in wireless ad hoc network routing protocols," in *ACM workshop on Wireless security*, 2003, pp. 30–40.

[40] Y.-A. Huang and W. Lee, "A cooperative intrusion detection system for ad hoc networks," in *the 1st ACM workshop on Security of ad hoc and sensor networks*, 2003, pp. 135–147.



Qijun Gu is an assistant professor at the Department of Computer Science, Texas State University - San Marcos. He received the Ph.D. degree in Information Sciences and Technology from Pennsylvania State University in 2005, the Master degree and the Bachelor degree from Peking University, China, in 2001 and 1998. His research interests include vulnerability in sensor applications, authentication in ad hoc and sensor networks, and security in peer to peer systems.

Jawad Drissi is an assistant professor at the Department of Computer Science, Texas State University - San Marcos. He received the Master degree (1995) and the Ph.D.degree (2000) in computer science from Montreal University. He also received the Master degree in Mathematics from Grenoble University, France, in 1981. His research interests include specification, verification and testing of communication protocols; network security and distributed systems.