This full text paper was peer reviewed at the direction of IEEE Communications Society subject matter experts for publication in the IEEE Globecom 2011 proceedings.

Lightweight Source Anonymity in Wireless Sensor Networks

Phillip Reindl[1], Xiaojiang Du[2], Kendall Nygard[1], Hongli Zhang[3]

[1]Dept. of Computer Science, North Dakota State University, Fargo, ND, 58105, {phillip.reindl, kendall.nygard}@ndsu.edu

[2]Dept. of Computer and Information Sciences, Temple University, Philadelphia, PA, 19122, dux@temple.edu

[3]School of Computer Science and Technology, Harbin Institute of Technology, China, zhanghongli@hit.edu.cn

***Abstract* – In many applications of Wireless Sensor Networks (WSN), the source of an event needs to be protected. In resource constrained WSN, providing source anonymity is a challenging task. A traditional approach for hiding source in WSN is to let all nodes generate dummy data packets even if they have no event to report. However, this kind of approach introduces large overhead. In order to reduce the large overhead of sending dummy data packets, we propose using a much shorter control packet to achieve source anonymity. The short control packets are used to coordinate the transmissions of dummy data packets, which prevent revealing the source node and hence provides source anonymity in WSN. We evaluate the performance of our anonymity scheme via ns-2 simulations. The simulations show that our scheme has much less traffic overhead than an existing anonymity scheme.**

Keywords - anonymity; wireless sensor networks; security; dummy traffic

I. INTRODUCTION

Wireless Sensor Networks (WSN) have many applications. In some network scenarios (such as military battlefield surveillance), identifying the physical source of a message represents a security breach, even if the message content is not revealed. This is a particular threat as it is relatively easy for an adversary to trace network traffic that uses wireless communications.

A number of literatures (e.g., [1] - [5]) have studied source anonymity in WSN. In [1], Shao *et al.* present the FitProbRate (FPR) scheme. FPR considers a homogeneous sensor network and an outside attacker performing traffic analysis. End-to-end encryption is used so intermediate sensors are unable to determine whether a given packet is dummy or data traffic. The main contribution of [1] is the statistically strong anonymity scheme with minimal delay for event data. All nodes send data traffic with a random delay of mean λ. When a node has event data to send, it computes the minimum delay that fits the probability distribution within a given confidence threshold. Shao *et al.* find that the average delay incurred is approximately $\lambda/10$.

In [2], Yang *et al.* extend the FPR scheme by introducing proxy nodes that filter the dummy traffic, reducing the overall network energy dissipation. This method is still expensive, as the dummy messages have the same length as the event data messages. Furthermore, the proxy nodes are bottlenecks of the system and if one fails then it can disrupt a large area of the network.

Ahn *et al.* introduce the concepts of *k*-anonymity in [3]. *k*-anonymity means that an adversary can only narrow the node of interest to within a set of *k* nodes. Thus, in a network of *n* nodes, full anonymity is achieved when $k = n$. They design a scheme that gives both the sender and the receiver *k*-anonymity. By limiting the anonymity to a subset of *n*, anonymity acceptable for many applications is achieved at much lower cost. Their scheme relies on public key encryption, which requires heavy computations and hence it is unsuitable for many sensor networks.

The approach for efficient source anonymity in sensor networks taken by [4] and [5] is to send data packets on a random walk before routing them to the base station. The random walk sends the data to an intermediate destination before final routing to the base station. This layer of re-direction may add a degree of security, but is not robust against a global observer.

Kong *et al.* [6] present ANODR, which provides route anonymity and location privacy for ad-hoc networks. Route pseudonyms are employed which require a costly set-up phase. The heavy burdens of trap-door encryption and route setup make the scheme unsuitable for energy-constrained sensor networks. Furthermore, the mixing techniques employed by ANODR require constant traffic among varied sources and destinations, where sensor networks have a fixed destination (the base station).

In this research, we propose a novel idea for source anonymity by adding short *control packets* that coordinate the transmission of longer, less frequent data packets. An observer will only know that an event has occurred, but will not be able to find out what the event was or where it happened. We compare the performance of our scheme and the FPR scheme [1] by using the ns2 network simulator [10]. Transmission overhead includes all traffic (except sensing data) that is routed to the base station. Latency is the amount of time between when the sensor event occurs and the data arrives at the base station. Our scheme seeks to minimize both latency and transmission overhead.

II. ATTACK AND NETWORK MODEL

The scenario described in [1] and [2] is a sensor network monitoring endangered animals such as giant pandas. The attacker is a hunter who has placed a sensor network in the same area that can monitor radio transmissions. We are thus concerned with a global, passive observer. In this scenario, the hunter's goal is to use traffic analysis to find the source of an event data, which is where the panda is located.

978-1-4244-9268-8/11/$26.00 ©2011 IEEE

Due to cost reason, inexpensive small sensor nodes do not have tamper-resistant hardware, and may be compromised by an adversary [7]. Similar to the assumption in [8], we assume that there is an initial secure period where all nodes are trusted and cryptographic keys are shared among neighboring nodes. After the secure period, some sensors in the network may be compromised and may behave arbitrarily.

Each sensor node shares a pair-wise key with the base station (BS). This key is used to encrypt data packets sent from the sensor to the BS. Each sensor has a broadcast key known by its neighbors. This key is used for local broadcasts. Finally, every sensor shares a pair-wise key with each of its neighbors. All radio links are bidirectional. A Medium Access Control (MAC) layer protocol is in place to coordinate communications among neighboring nodes. In all cases, transmissions are encrypted such that an outside observer cannot see the message contents.

III. The Anonymity Scheme For WSN

A. Scheme Description

A sensor network consists of many small sensor nodes. In order to provide source anonymity, dummy packets are sent by nodes that do not have data to send. If all (or most) dummy packets have the same length as a data packet (this is the case for the scheme in [1]), then it is very costly. In this paper, we propose an anonymity scheme that uses short control packets to coordinate (long) data packet transmissions. The scheme is referred to as the Control Packet based Anonymity (CPA) scheme, and it is presented in Fig. 1. In the following, we describe the CPA scheme.

Nodes transit between two states: *Idle* and *Ready*. *Idle* nodes periodically send 'no' *control packets* indicating that they have no data to send. A node (say *i*) with data sends out 'yes' *control packets*, and will send a data packet around a specified time *t*. Around the time *t*, node *i* sends out its event data packet, and nodes without data send a dummy 'data' packet (the same length as the real data packet). A node moves to the *Ready* state when one of two conditions are satisfied:

1) It has data to send.
2) It has received a '*yes*' control packet.

If a node has data, it will send out '*yes*' control packets, which are forwarded to all other nodes in the network, so soon all nodes in the network move to the *Ready* state.

After transiting from *Idle* to *Ready*, a node sets a timer to send a (real or dummy) data packet. When the timer fires, the node will send either a real or dummy data packet. The node will then transit back to *Idle*. The sequence of transitions from *Idle* to *Ready*, transmission of a data packet, and transitioning back to *Idle* is referred to as a cycle. The cycles of the nodes in the network are synchronized using control packets. Fig. 1 contains a formal description of the scheme.

Nodes send out control packets with a random delay following an exponential probability function. The exponential parameter λ gives the mean time between transmissions. Nodes maintain the history of inter-message time. In Fig. 1, the function *MinDelay* returns a small delay that fits the exponential distribution with the given mean and history. A statistical test is used to verify fit to a desired confidence. The function *RecoverMean* returns a delay that restores the proper mean. After transiting to *Ready* state, a node sends the first *'yes'* control packet with a delay given by *MinDelay*. Subsequent control packets are sent with a delay given by the function *RecoverMean*.

Input: Distance H the node is from the far edge of the network (Hops), Mean delay between control packets λ

1. *Tp* :=
2. φ := {}
3. *hasData* := FALSE
4. *state* := IDLE

IDLE state

→ 'no' Control Packet received
1. *state* := IDLE

→ 'yes' Control Packet received
1. *ctrlDelay* := MinDelay(φ, λ)
2. **set** ControlTimer
3. *timeToSendData* := packet[tsd]
4. *dataDelay* := packet[tsd] - CurrentTime
5. **set** DataTimer
6. *state* := READY

→ ControlTimer fired
1. φ := φ ∪ {*ctrlDelay*}
2. **send** 'no' Control Packet
3. *ctrlDelay* := RecoverMean(φ, λ)
4. **set** ControlTimer
5. *state* := IDLE

→ Sensor Event
1. **set** *hasData* := TRUE
2. *ctrlDelay* := MinDelay(φ, λ)
3. **set** ControlTimer
4. *timeToSendData* := *Tp* + CurrentTime
5. *dataDelay* := *Tp*
6. *state* := READY

READY state

→ 'no' Control Packet received
→ 'yes' Control Packet received
1. *state* := READY

→ ControlTimer fired
1. φ := φ ∪ {*ctrlDelay*}
2. **send** 'yes' Control Packet
3. *ctrlDelay* := RecoverMean(φ, λ)
4. **set** ControlTimer
5. *state* := READY

→ DataTimer fired
1. **if** *hasData* = TRUE **then**
1. **send** 'Event' Data Packet
2. **else**
1. **send** 'Dummy' Data Packet
3. **endif**
4. *hasData* := FALSE
5. *state* := IDLE

→ Sensor Event
1. **set** *hasData* := true
2. *state* := READY

Fig. 1: The control packet based anonymity scheme

Assume all the sensor clocks are synchronized. Each node has an estimation, TP, for how long its control packet propagate through the network. The *Time to Send Data* (TSD) field is set to the current time + TP. An *idle* node that receives a *'yes'* control packet will use the TSD from the incoming *'yes'* packet.

If TP is overestimated, then nodes will wait longer than necessary for the '*yes*' control packets to propagate through the network, causing increased latency for the data packet. On the other hand, if TP is underestimated, then the '*yes*' control packets will not have time to propagate through the network, and some nodes will not send dummy traffic. This will give the adversary some clues to the event location.

Shao *et al.* found that the FitProbRate [1] scheme reduced latency from 10.87s for the constant rate scheme to under 1s using *MinDelay* function. Based on the result in [1], a good estimation for the TP is (λ/10)H, where H is the number of hops between a node and the far edge of the network.

When the time matches the TSD, all *Ready* state nodes (most or all nodes in the network) send a real/dummy data packet to the base station. This hides the identity of the real source node and provides anonymity for the traffic source. Even a global observer would not be able to find out the real source of the event. After that, all sensors reset to the *Idle* state, and the next cycle starts.

B. Securing Control Packets

Data packets are much longer than control packets. We use 30 bytes as the data packet length, as this is a standard data packet length in existing systems such as TinyOS [9]. The contents of the data packet is highly application dependent, so we make no assumptions other than the size of the data packet. Control packets are very short, containing only the necessary information such as the node id, a bit indicating whether there is data to send, and *Time to Send Data*. The format of a control packet is given in Fig. 2. The control packet has four fields: PrevHop, pseudo-ID, Ready, TimeToSendData.

PrevHop	KB*j*{pseudo-ID | Ready | TimeToSendData}

Fig. 2: Control packet format

PrevHop: 15-bit field containing the ID of sensor *j*, who is the current sender of this packet. This is needed so that neighboring sensors are able to use the correct broadcast key KB*j* to decrypt the remainder of the packet. 15-bit allows 32K unique node ids, which is sufficient for most sensor networks.

pseudo-ID: 15-bit field containing the pseudo node ID of the original source node that generated the data packet and sent out the first '*yes*' control packet of this cycle. The pseudo-ID is computed using a one-way hash function *f* as follows:

$$\text{pseudo-ID} = f(ID_S, C_S, K_{S,BS}) \qquad (1)$$

ID_S The true ID of the source node S

C_S A monotonically increasing sequence number for the '*yes*' control packets initiated by node S.

$K_{S,BS}$ Symmetric key shared by source node S and the BS.

Since *f* depends on the key $K_{S,BS}$ known only by S and the BS, only the base station is able to find out the ID_S (the real node ID) that is used to generate the pseudo-ID. The BS will verify the pseudo-ID and take action if fraud is detected. See subsection C for details.

Ready: 1-bit field. '1' means this is a '*yes*' control packet, '0' means this is a '*no*' control packet.

- *Ready* nodes will generate '*yes*' control packets, with the TimeToSendData field set to the scheduled time.
- *Idle* nodes will set their TSD to the TimeToSendData field of a '*yes*' packet they receive, and become *Ready*.

TimeToSendData: 24-bit field containing the expected time t_E to send data packets. The purpose of the control packet is to synchronize the transmission of real/dummy data packets among all nodes. In order to avoid collisions if all nodes transmit real/dummy data packets simultaneously, each node sends its data packet at a random time given by a Gaussian distribution with mean t_E.

If the TimeToSendData field specifies the time in seconds, a network lifetime of 194 days is supported before the clocks overflow. The total length of a control packet is 15 bits + 15 bits + 1 bit + 24 bits = 55 bits, or 7 bytes, which is much shorter than a data packet (30 bytes). Furthermore, the above length supports 32K unique sensor nodes and 194 days of operations. If the size or running time of a sensor network is smaller, one could reduce the length of a control packet. For example, for a sensor network with 1,000 nodes, we only need 10 bits for both the PrevHop and Pseudo-ID fields, which reduces the control packet size to only 45 bits.

C. Control Packet Verification

The BS verifies each control packet. The BS tracks C_S for every sensor in the network. The BS pre-computes the next pseudo-ID for each sensor by using equation (1), ID_S, C_S, and $K_{S,BS}$. When a '*yes*' control packet arrives, the BS compares the received pseudo-ID with the pre-computed pseudo-IDs. If a match is found, then the corresponding sensor node is the source of the control packet, and BS increases the C_S of this sensor, and pre-computes the next pseudo-ID. If no match is found, the control packet is deemed fraudulent. To provide fault tolerant to lost control packets (i.e., did not reach the BS), the BS may pre-computes several pseudo-IDs for each sensor, based on the current C_S, C_S+1, and C_S+2. As long as the received pseudo-ID matches one of the pseudo-IDs, it is considered ok. The control packet verification algorithm is illustrated in Fig. 3, where the parameter ω is the number of pre-computed pseudo-IDs. Large values of ω provider better tolerance to packet lost, but requires more BS storage and computations.

```
Input: pseudo-ID to test
1.      foreach L-node n
1.          for seq := n.last to n.last + ω
1.              if f(n.ID, seq, n.key) = pseudo-ID then
1.                  n.last := seq
2.                  return PASS
2.              endif
2.          endfor
2.      endforeach
3.      return FAIL
```

Fig. 3: The control packet verification algorithm

A compromised node is not expected to behave correctly. The above anonymity scheme is subject to the following resource depletion attack: A corrupt node may send false '*yes*' control packets. Since they are not verified by intermediate sensors before they forward the control packet to other nodes, a single corrupt node may force the entire network to send

unnecessary dummy data packets, which wastes a lot of resources.

To defend the above attack, each sensor keeps record of two things: (a) pseudo-IDs in recent '*yes*' control packets, and (b) the upstream neighbor who forwarded each control packet. In the event that the BS detects a fraudulent pseudo-ID (based on equation (1)), the BS can query the network to find out who was the source of the '*yes*' control packet. This can be done by tracing (backwards) the pseudo-ID and the upstream neighbor at each step.

IV. Performance Analysis

In this Section, we compute the total amount of network traffic and the latency of the two anonymity schemes - FPR and CPA.

A. Notations

We list the mathematic notations below:

Φ - Network lifetime in seconds.

Ω - The number of events during the network lifetime.

α - The length of a control packet in bytes.

β - The length of a data packet in bytes.

n - The number of sensors in the network.

We are only interested in the additional latency caused by the CPA scheme. In the following analysis, we do not include latencies caused by queuing, MAC contention and routing delays. However, our ns2 [10] simulations (Section V) do include the above delays.

B. The Performance of FPR

Suppose the average distance of a sensor to the BS is d hops. Under FPR, each node sends out a dummy/real data packet per λ seconds, independent of whether it has data to send. Each packets is routed via an average of d hops to reach the BS. The network lifetime is Φ, so each of n nodes sends $\frac{\Phi}{\lambda}$ packets over the network lifetime. Every packet gets retransmitted at each of d hops until it reaches the BS. Each packets is β bytes in length. Hence, the total amount of traffic generated in the network using FPR is $n\frac{\Phi}{\lambda}d\beta$ bytes.

When a node has data to send, the data is sent directly to the BS with a delay given by the *MinDelay* function. According to [1], the average latency of FPR is $\frac{\lambda}{10}d$ seconds.

C. The Performance of CPA

Sensors are an average of d hops from the BS and h hops from the far edge of the network.

A node sends a data packet to the BS for each detected event. Each data packet is relayed by an average of d sensors. Each data packet is β bytes. Hence, the total amount of real data traffic is $\Omega d\beta$ bytes. Recall that for each real data packet, all sensors (except the real source node) will send out a dummy data packet. Hence, the total amount of real+dummy data traffic is $n\Omega d\beta$ bytes. Now let's calculate the total amount of control traffic. For each real event, a control packet of length α will be generated to synchronize all nodes in the network, and the control packet is broadcasted in the network (i.e., relayed by n nodes). The total number of events is Ω. Hence, the total amount of control traffic is $n\alpha\Omega$ bytes. And the total amount of traffic in the network is $n\Omega d\beta + n\alpha\Omega$ bytes.

The average latency is $\frac{\lambda}{10}h$ seconds. This is the time it takes for a 'yes' control packet to propagate through the network. When the data timer fires, all nodes send their data packets with any no further delay. Under the CPA scheme, the bulk of the latency is a result of the control packet propagation.

V. Experimental Evaluation

A. Experimental Setup

In order to evaluate the efficiency of our CPA scheme, we implemented both the FPR and CPA schemes in the ns2 network simulator [10], and compared their performances. We fixed the network size, and run simulations for different total numbers of sensors, from 25 sensors to 275 sensors. The random number function of ns2 was utilized with predefined seed values to ensure repeatable "random" deployments. In all cases, the BS was located in the upper left corner of the placement area.

The mean time between events σ was varied over the set {5, 10, 20, 30, 45, 60, 90, 120, 180, 1000} (in seconds). When σ=1000, no events occur during the simulation. This gives a baseline result for the traffic generated by the schemes when no events happen. All simulations were run for 600 seconds, and 10 runs were made with different random number seeds. The results presented below are the average of the 10 runs.

B. Simulation Results

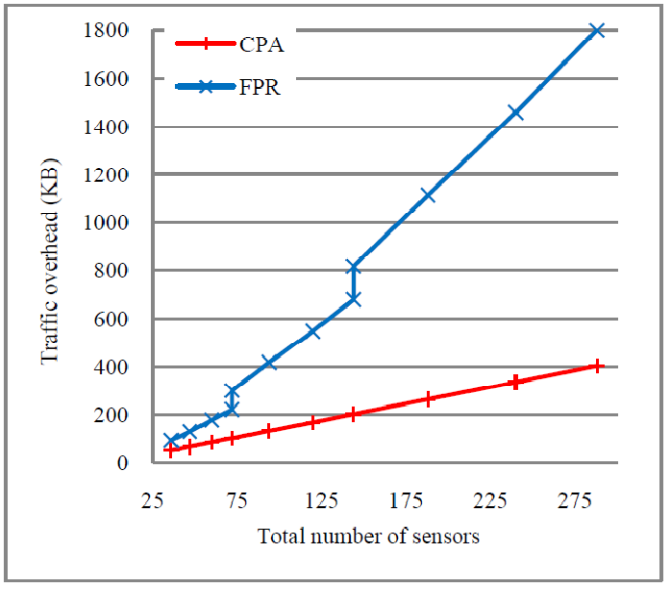

Fig. 4: Traffic overhead vs. number of sensor nodes

Fig. 4 compares the traffic overhead of our CPA scheme and the FPR anonymity scheme [1], for different number of sensor nodes (i.e., network density). Fig. 4 shows that the overhead of our CPA scheme is much smaller than that of FPR, especially for large number of sensor nodes. This is because FPR sends out long dummy packets for anonymity, and this causes very large overheads when there are more sensor nodes in the network, since more nodes means more (long) dummy packets. Our CPA scheme only uses short control packets, so the overhead only increases slightly when the number of nodes increases.

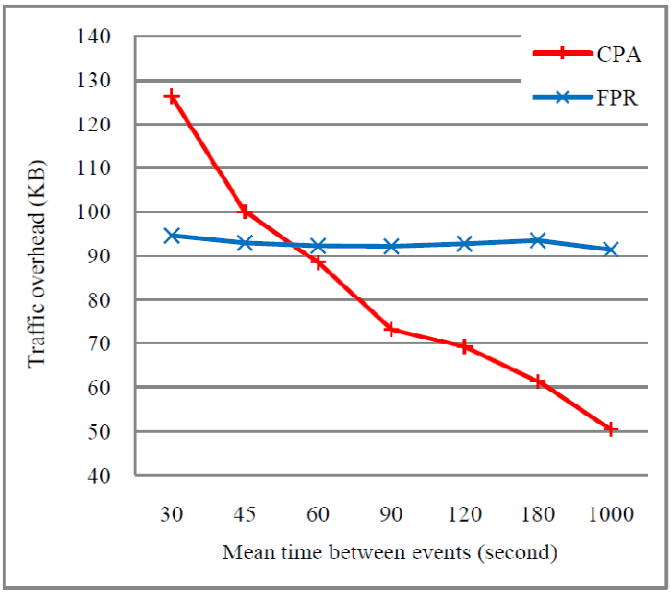

Fig. 5: Simulation results

Another important parameter is the expected mean time between events - σ. Fig. 5 shows traffic overheads of CPA and FPR for different σ. FPR has an almost constant traffic overhead, regardless of how frequent the data is generated. CPA is a reactive scheme, and it only generates control packets when there is a data packet to be sent. Fig. 5 shows that CPA has more traffic overhead than FPR when the data events are frequent (i.e., for small σ), but CPA has much less overhead than FPR when the event frequency is not high (i.e., for large σ). Hence, CPA is well suited for sensor networks with not-so-frequent events, which is the case for many sensor networks. Recall that many sensor nodes/networks are designed to operate in low duty cycle, because the data events are not frequent.

We also measured the latency of the FPR and the CPA schemes using the ns-2 simulations. Our results showed that both schemes have similar latency.

VI. Conclusions

In this paper, we proposed an efficient scheme to provide event source anonymity for sensor networks. Our scheme significantly reduces the amount of dummy traffic by using short control packets (instead of long dummy packets) to synchronize the transmissions of long real/dummy data packets. A major concern for efficient source anonymity is the latency imposed by the probabilistic delay used to thwart traffic analysis. By using short control packets to coordinate the transmission of dummy/real data packets, a much smaller probabilistic delay can be achieved with less traffic overhead. The ns2 simulation results showed that our anonymity scheme has much less traffic overhead than the FPR scheme.

Acknowledgment

This research was supported in part by the US National Science Foundation under grants CNS-0963578, CNS-1002974, CNS-1022552, and CNS-1065444, as well as the US Army Research Office under grant W911NF-08-1-0334.

References

[1] M. Shao, Y. Yang, S. Zhu, and G. Cao, "Towards Statistically Strong Source Anonymity for Sensor Networks," in *Proc. of the IEEE INFOCOM 2008*, pp. 51-55, 2008.

[2] Y. Yang, M. Shao, S. Zhu, B. Urgaonkar, and G. Cao, "Towards Event Source Unobservability with Minimum Network Traffic in Sensor Networks," in *Proc. of the ACM WiSec*, Mar. 2008, pp. 77-88.

[3] L. Ahn, A. Bortz, and N. Hopper, "k-Anonymous Message Transmission," in *Proc. of the ACM CCS*, pp. 122-130, 2003.

[4] P. Kamat, Y. Zhang, W. Trappe, and C. Ozturk, "Enhancing Source-Location Privacy in Sensor Network Routing," in *Proc. of the IEEE ICDCS*, pp. 599-608, 2005.

[5] M. Shao, W. Hu, S. Zhu, G. Cao, S. Krishnamurth, and T. La Porta, "Cross-Layer Enhanced Source Location Privacy in Sensor Networks," in *Proc. of the IEEE SECON '09*, pp. 1-9, 2009.

[6] J. Kong, and X. Hong, "ANODR: anonymous on demand routing with untraceable routes for mobile ad-hoc networks," in *Proc. of the ACM MobiCom*, pp. 291-302, 2003.

[7] Y. Yang, X. Wang, S. Zhu, and G. Cao, "Distributed Software-based Attestation for Node Compromise Detection in Sensor Networks," in *Proc. of the IEEE International Symposium on Reliable Distributed Systems (SRDS)*, Oct. 2007., pp. 219-230, 2007.

[8] S. Zhu, S. Setia, and S. Jajodia, "LEAP: Efficient Security Mechanisms for Large-Scale Distributed Sensor Networks," in *Proc. of the ACM CCS*, pp. 62-72, 2003.

[9] C. Karlof, N. Sastry, and D. Wagner, "TinySec: a link layer security architecture for wireless sensor networks," in *Proc. of the 2nd International Conference on Embedded Networked Sensor Systems*, pp. 162-175, 2004.

[10] The Network Simulator: ns-2, http://www.isi.edu/nsnam/ns/