Temporal Privacy in Wireless Sensor Networks

Pandurang Kamat, Wenyuan Xu, Wade Trappe, Yanyong Zhang
Wireless Information Network Laboratory (WINLAB), Rutgers University
Email: {pkamat, wenyuan, trappe, yyzhang}@winlab.rutgers.edu

Abstract

Although the content of sensor messages describing "events of interest" may be encrypted to provide confidentiality, the context surrounding these events may also be sensitive and therefore should be protected from eavesdroppers. An adversary armed with knowledge of the network deployment, routing algorithms, and the base-station (data sink) location can infer the temporal patterns of interesting events by merely monitoring the arrival of packets at the sink, thereby allowing the adversary to remotely track the spatio-temporal evolution of a sensed event. In this paper, we introduce the problem of temporal privacy for delay-tolerant sensor networks and propose adaptive buffering at intermediate nodes on the source-sink routing path to obfuscate temporal information from an adversary. We first present the effect of buffering on temporal privacy using an information-theoretic formulation and then examine the effect that delaying packets has on buffer occupancy. We evaluate our privacy enhancement strategies using simulations, where privacy is quantified in terms of the adversary's estimation error.

1. Introduction

Sensor networks are being increasingly deployed to collect measurements around a vast array of phenomena. Conventional security services, such as encryption and authentication, have been migrated to the sensor domain [15] to keep these measurements confidential. Despite this there are many aspects associated with the creation and delivery of sensor messages that remain unprotected by conventional security mechanisms. Protecting such contextual information, which is as important as protecting the content of sensor messages, thus demands complementary techniques [8, 11, 14].

Among a broad range of contextual information that a sensor network should protect, where and when an asset was observed, is particularly sensitive, especially for sensor networks that monitor high-value, mobile targets or assets. In order to prevent such spatio-temporal information from being leaked, the underlying sensor network must ensure the privacy of the following two types of information: (1) the location of the source node(s) that observed the target, and (2) the time when the source node(s) observed the target. The importance of source location privacy, as well as techniques to achieve this privacy, has been extensively studied in [11, 14]. The protection of temporal information, a problem which we refer to as *temporal privacy* in this paper, however, has received little attention so far.

Protecting the temporal privacy of a sensor network is a challenging issue, particularly as the concept of temporal privacy has not yet been formally defined. In this paper, we address this need by providing a formal definition of temporal privacy that is built upon information theoretic concepts. Specifically, if we assume that the adversary stays at the sink, collects all the packets that are generated by a source, and tries to infer the creation times of these packets from the times when they are received, then the temporal privacy can be defined as the mutual information between the received time sequences and the creation time sequences. In order to minimize the mutual information, we propose to buffer each packet at intermediate nodes along the routing path between a source sensor and the sink. The insights from the information-theoretic study further reveal that random delays that follow an exponential distribution will better protect temporal privacy than other distributions.

Buffering packets at intermediate nodes can protect temporal privacy, but it may lead to the requirement of large amount of buffer space at each node, especially for large-scale sensor networks as considered in our study. As a result, we have also studied the buffer demands at each node using a queuing formulation, and found that the buffer demands can become rather high as the network scales. Considering the fact that sensor nodes usually have serious resource constraints, including available buffer space, we have devised an adaptive buffering strategy that preempts buffered packets to accommodate newly arriving packets if the buffer is full.

We begin the paper in Section 2 by describing our sensor network model, overview the problem of temporal privacy and how additional buffering can enhance privacy. We then examine the two conflicting aspects of buffering: in Section 3, we formulate temporal privacy from an information-theoretic perspective, and in Section 4, we examine the stress that additional delay places on intermediate buffers. Then, in Section 5, we present an adaptive buffering strategy that effectively manages these tradeoffs through the preemptive release of packets as buffers attain their capacity and evaluate its performance through simulations. Finally, we present related work in Section 6, and conclude the paper in Section 7.

2. Temporal Privacy in Sensor Networks

We start our overview by describing a couple of scenarios that illustrate the issues associated with temporal privacy. To begin, consider a sensor network that has been deployed to monitor an animal habitat [11, 16]. In this scenario, animals ("assets") move through the environment, their presence is sensed by the sensor network and reported to the network sink. The fact that the network produces data and sends it to the sink provides an indication that the animal was present at the source at a specific time. If an adversary is able to associate the origin time of the packet with a sensor's location, then the adversary will be able to track the animal's behavior– a dangerous prospect if the animal is endangered and the adversary is a hunter! This same scenario can be easily translated to a tactical environment, where the sensor network monitors events in support of military networked operations. In asset tracking, if we add temporal ambiguity to the time that the packets are created then, as the asset moves, this would introduce spatial ambiguity and make it harder for the adversary to track the asset.

Temporal privacy amounts to preventing an adversary from inferring the time of creation associated with one or more sensor packets arriving at the network sink. In order to protect the temporal context of the packet's creation, it is possible to introduce additional, random delay to the delivery of packets in order to mask a sensor reading's time of creation. Although delaying packets might increase temporal privacy, this strategy also necessitates the use of buffering either at the source or within the network and places new stress on the internal store-and-forward network buffers.

The sensor network model that we use involves:

Delay-Tolerant Application: A sensor application that is delay-tolerant, but not entirely delay insensitive, in the sense that observations can be delayed by reasonable amounts of time before arriving at the monitoring application, thereby allowing us to introduce additional delay in packet delivery.

Encrypted Payload: The payload contains application-level information, such as the sensor reading, application sequence number, and the time-stamp associated with the sensor reading. Conventional encryption is used to protect the sensor application's data.

Cleartext Headers: The headers associated with essential network functionality are not encrypted. For example, the routing header associated with [18], and used in the TinyOS 1.1.7 release (described in `MultiHop.h`) includes the ID of the previous hop, the ID of the origin (used in the routing layer to differentiate between whether the packet is being generated or forwarded), the routing-layer sequence number (used to avoid loops, not flow-specific and hence cannot help the adversary in estimating time of creation), and the hop count.

The assumptions that we have for the adversary are

Deployment-Aware: By Kerckhoff's Principle [17], we assume the adversary has knowledge of the networking and privacy protocols being employed by the sensor network. In particular, the adversary knows the delay distributions being used by each node in the network. Further, we assume the adversary has knowledge of the positions of all sensor nodes in the network.

Able to Eavesdrop: The adversary is able to eavesdrop on communications in order to read packet headers, or control traffic. We emphasize that the adversary is not able to decipher packet contents by decrypting the payloads, and hence the adversary must infer packet creation times solely from network knowledge and the time it witnesses a packet.

Non-intrusive: The adversary does not interfere with the proper functioning of the network, otherwise intrusion detection measures might flag the adversary's presence. In particular, the adversary does not inject or modify packets, alter the routing path, or destroy sensor devices.

These security assumptions are intended to give the adversary significant power and thus, if our temporal privacy techniques are robust under these assumptions, they can be considered powerful under more general threat conditions.

2.1. The Baseline Adversary Model

Our sensor network model assumes multiple source nodes that create packets and send these packets to a common sink via multi-hop networking. The adversary stays at the sink, observes packet arrivals, and estimates the creation times of these packets. We note that, while it may seem like the adversary would be better off being mobile or that the adversary be located at several random places within the network, it is not so. *Since all activities in a sensor network are reported to the sink, being closer to the sink enables the adversary to maximize his chances of observing as many traffic flows as possible.*

To better focus on temporal privacy of a sensor network, we assume a rather powerful adversary that can acquire the following information about the underlying network:

1. The hop count h_i, of flow i. This can be inferred by the adversary by looking at hop-count information in the packet headers.
2. The transmission delay on a node, τ.

For an observed packet arrival time z, our adversary estimates the creation time of this packet as $x' = z - h\tau$. In the literature, the *square error* is often used to quantify the estimation error, i.e. $(x' - x)^2$ where x is the true creation time. Similarly, for a series of packet arrivals from the same flow

$z_1, z_2, \ldots, z_m$, our adversary estimates their creation times as $x'_1, x'_2, \ldots, x'_m$, and $x'_i = z_i - h\tau$. The total estimation error for m packets is then calculated as the *mean square error* $MSE = \sum(x'_i - x_i)^2/m$. A network that causes an adversary to have a higher estimation error consequently better preserves the temporal privacy of the source.

3. Temporal Privacy Formulation

We start by first examining the theoretical underpinnings of temporal privacy. Our discussion starts by first setting up the formulation using a simple network of two nodes transmitting a single packet, and then we extend the formulation to more general network scenarios.

3.1. Two-Party Single-Packet Network

We begin by considering a simple network consisting of a source S, a receiver node R, and an adversarial node E that monitors traffic arriving at R. The goal of preserving temporal privacy is to make it difficult for the adversary to infer the time when a specific packet was created. Suppose that the source sensor S observes a phenomena and creates a packet at some time X. In order to obfuscate the time at which this packet was created, S can choose to locally buffer the packet for a random amount of time Y before transmitting the packet. Disregarding the negligible time it takes for the packet to traverse the wireless medium, both R and E will witness that the packet arrives at a time $Z = X + Y$. The legitimate receiver can decrypt the payload, which contains a timestamp field describing the correct time of creation. The adversary's objective is to infer the time of creation X, and since it cannot decipher the payload, it must make an inference based solely upon the observation of Z and (by Kerckhoff's Principle) knowledge of the buffering strategy employed at S.

The ability of E to infer X from Z is controlled by two underlying distributions: first, is the a priori distribution $f_X(x)$, which describes the knowledge the adversary had for the likelihood of the message creation prior to observing Z; and second, the delay distribution $f_Y(y)$, which the source employs to mask X. In classical security and privacy, the amount of information that E can infer about X from observing Z is measured by the mutual information [6, 17]:

$$I(X;Z) \;=\; h(X) - h(X|Z) = h(Z) - h(Y) \quad (1)$$

where $h(X)$ is the differential entropy of X. We note that, due to the relationship between mutual information and mean square error [10], large $I(X;Z)$ implies that a well-designed estimator of X from Z will have small MSE. For certain choices of f_X and f_Y, we may directly calculate $I(X;Z)$. For general distributions, the entropy-power inequality [6] gives a lower bound

$$I(X;Z) \geq \frac{1}{2\ln 2}\left(2^{2h(X)} + 2^{2h(Y)}\right) - h(Y). \quad (2)$$

In general, however, the distribution for X is fixed and determined by an underlying physical phenomena being monitored by the sensor. Since the objective of the temporal privacy-enhancing buffering is to hide X, we may formulate the temporal privacy problem as

$$\min_{f_Y(y)} I(X;Z) = h(X+Y) - h(Y),$$

or in other words, choose a delay distribution f_Y so that the adversary learns as little as possible about X from Z.

3.2. Two-Party Multiple-Packet Network

We now extend the formulation of temporal privacy to the more general case of a source S sending a stream of packets to a receiver R in the presence of an adversary E. In this case, the sender S will create a stream of packets at times $X_1, X_2, \ldots, X_n, \ldots$, and will delay their transmissions by $Y_1, Y_2, \ldots, Y_n, \ldots$. The packets will be observed by E at times $Z_1, Z_2, \ldots, Z_n, \ldots$. One delay strategy would have packets released in the same order as their creation, i.e. $Z_1 < Z_2 < \ldots < Z_n$, which would correspond to choosing Y_j to be at least the wait time needed to flush out all previous packets. Such a strategy does not reflect the fact that most sensor monitoring applications do not require that packet ordering is maintained. Therefore, a more natural delay strategy would involve choosing Y_j independent of each other and independent of the creation process $\{X_j\}$. Consequently, there will not be an ordering of $(Z_1, Z_2, \ldots, Z_n, \ldots)$.

In our sensor network model, however, we assumed that the sensing application's sequence number field was contained in the encrypted payload, and consequently the adversary does not directly observe $(Z_1, Z_2, \ldots, Z_n, \ldots)$, but instead observes the sorted process $\{\tilde{Z}_j\} = \Upsilon(\{Z_j\})$, where $\Upsilon(\{Z_j\})$ denotes the permutations needed to achieve a temporal ordering of the elements of the process $\{Z_j\}$, i.e. $\{\tilde{Z}_j\} = (\tilde{Z}_1, \tilde{Z}_2, \ldots, \tilde{Z}_n, \ldots)$ where $\tilde{Z}_1 < \tilde{Z}_2 < \cdots$. The adversary's task thus becomes inferring the process $\{X_j\}$ from the sorted process $\{\tilde{Z}_j\}$. The amount of information gleaned by the adversary after observing $\tilde{Z}^n = (\tilde{Z}_1, \cdots, \tilde{Z}_n)$ is thus $I(X^n; \tilde{Z}^n)$, and the temporal-privacy objective of the system designer is to make $I(X^n; \tilde{Z}^n)$ small.

Although it is analytically cumbersome to access $I(X^n; \tilde{Z}^n)$, we may use the data processing inequality [6] on $X^n \rightarrow Z^n \rightarrow \tilde{Z}^n$ to obtain the relationship $0 \leq I(X^n, \tilde{Z}^n) \leq I(X^n, Z^n)$, which allows us to use $I(X^n, Z^n)$ in a pinching argument to control $I(X^n, \tilde{Z}^n)$. Expanding $I(X^n, Z^n)$ as

$$I(X^n, Z^n) = h(Z^n) - h(Y^n) = \sum_{j=1}^{n} I(X_j, Z_j) \quad (3)$$

we may thus bound $I(X^n, Z^n)$ using the sum of individual mutual information terms.

As before, the objective of temporal privacy enhancement is to minimize the information that the adversary gains, and hence to mask $\{X_j\}$, we should minimize $I(X^n, Z^n)$. Although there are many choices for the delay process $\{Y_j\}$, the general task of finding a non-trivial stochastic process $\{Y_j\}$ that minimizes the mutual information for a specific temporal process $\{X_j\}$ is challenging and further depends on the sensor network design constraints (e.g. buffer storage). In spite of this, however, we may seek to optimize within a specific type of process $\{Y_j\}$, and from this make some general observations.

As an example of this, let us look at an important and natural example. Suppose that the source sensor creates packets at times $\{X_j\}$ as a Poisson process of rate λ, i.e. the interarrival times A_j are exponential with mean $1/\lambda$, and that the delay process $\{Y_j\}$ corresponds to each Y_j being an exponential delay with mean $1/\mu$. We note that our choice of a Poisson source is intended for explanation purposes, and that more general packet creation processes can be handled using the same machinery. One motivation for choosing an exponential distribution for the delay is the well-known fact that the exponential distribution yields maximal entropy for non-negative distributions. We note that $X_j = \sum_{k=1}^{j} A_k$ (and hence the X_j are j-stage Erlangian random variables with mean j/λ). Using the result of Theorem 3(d) from [3], we have that

$$\begin{aligned} I(X_j; Z_j) &= I(X_j; X_j + Y_j) \\ &= \ln\left(1 + \frac{j\mu}{\lambda}\right) - D\left(f_{X_j+Y_j} \| f_{\overline{X}_j+Y_j}\right) \\ &\leq \ln\left(1 + \frac{j\mu}{\lambda}\right). \end{aligned}$$

Here, the $D(f\|g)$ corresponds to the divergence between two distributions f and g, while $\overline{X}$ is the mixture of a point mass and exponential distribution with the same mean as X, as introduced in [3]. Since divergence is non-negative and we are only interested in pinching $I(X^n; \tilde{Z}^n)$, we may discard this auxiliary term. Using the above result, we have that

$$I(X^n, Z^n) \leq \sum_{j=1}^{n} \ln\left(1 + \frac{j\mu}{\lambda}\right) \qquad (4)$$

Our objective is to make

$$0 \leq I(X^n; \tilde{Z}^n) \leq I(X^n, Z^n) \leq \sum_{j=1}^{n} \ln\left(1 + \frac{j\mu}{\lambda}\right)$$

small, and we can see that by tuning μ to be small relative to λ (or equivalently, the average delay time $1/\mu$ to be large relative to the average interarrival time $1/\lambda$), we can control the amount of information the adversary learns about the original packet creation times. It is clear that choosing μ too small will place a heavy load on the source's buffer.

3.3. Multihop Networks

In the previous subsection, we considered a simple network case consisting of two nodes, where the source performs all of the buffering. More general sensor networks consist of multiple nodes that communicate via multi-hop routing to a sink. For such networks, the burden of obfuscating the times at which a source node creates packets can be shared amongst other nodes on the path between the source and the sensor network sink. To explain, we may consider a generic sensor network consisting of an abundant supply of sensor nodes, and focus on an N-hop routing path between the source and the network sink. By doing so, we are restricting our attention to a line-topology network $S \to F_1 \to F_2 \to \cdots \to F_{N-1} \to R$, where R denotes the receiving network sink, and F_j denotes the j-th intermediate node on the forwarding path.

By introducing multiple nodes, the delay process $\{Y_j\}$ can be decomposed across multiple nodes as

$$Y_j = Y_{0j} + Y_{1j} + \cdots + Y_{N-1,j},$$

where Y_{kj} denotes the delay introduced at node k for the j-th packet (we use Y_{0j} to denote the delay used by the source node S). Thus, each node k will buffer each packet j that it receives for a random amount of time Y_{kj}.

This decomposition of the delay process $\{Y_j\}$ into sub-delay processes $\{Y_{kj}\}$ allows for great flexibility in achieving both temporal privacy goals and ensuring suitable buffer utilization in the sensor network. For example, it is well-known that traffic loads in sensor networks accumulate near network sinks, and it may be possible to decompose $\{Y_j\}$ so that more delay is introduced when a forwarding node is further from the sink.

4. Queuing Analysis

Although delaying packets might increase temporal privacy, such a strategy places a burden on intermediate buffers. In this section we will examine the underlying issues of buffer utilization when employing delay to enhance temporal privacy.

When using buffering to enhance temporal privacy, each node on the routing path will receive packets and delay their forwarding by a random amount of time. As a result, sensor nodes must buffer packets prior to releasing them, and we may formulate the buffer occupancy using a queuing model. In order to start our discussion, let us again examine the simple two-node case where a source node S generates packets according to an underlying process and the packets are delayed according to an exponential distribution with average delay $1/\mu$, prior to being forwarded to the receiver R. If we assume that the creation process is Poisson with rate λ (if the process is not Poisson, the source may introduce additional delay to shape the traffic or, at the expense of lengthy derivations, similar results can be arrived at using embedded Markov chains), then the buffering process can be viewed as an $M/M/\infty$ queue where, as new packets arrive at the buffer, they are assigned to a new "variable-delay server" that processes each packet according to an exponential distribution with mean $1/\mu$. Following the standard results for $M/M/\infty$ queues, we have that the amount of packets being stored at an arbitrary time, $N(t)$, is Poisson distributed, with $p_k = P\{N(t) = k\} = \frac{\rho^k}{k!}e^{-\rho}$, where $\rho = \lambda/\mu$ is the system utilization factor. $\overline{N}$, the expected number of messages buffered at S, is ρ.

The slightly more complicated scenario involving more than one intermediate node allows for the buffering responsibility to be divided across the routing path. A tandem queuing network is formed, where a message departing from node i immediately enters an $M/M/\infty$ queue at node $i+1$. Thus, the inter-departure times from the former generate the inter-arrival times to the latter. According to Burke's Theorem [4], the steady-state output of a stable $M/M/m$ queue with input parameter λ and service-time parameter μ for each of the m servers is in fact a Poisson process at the same rate λ when $\lambda < \mu$. Hence, we may generally model each node i on the path as an $M/M/\infty$ queue with average input message rate λ, but with average service-time $1/\mu_i$ (to allow each node to follow its own delay distribution).

In practice a sensor network will monitor multiple phenomena simultaneously, and consequently there will be multiple source-sink flows traversing the network. Consider a sensor network deployment where multiple sensors generate messages intended for the sink, and each message is routed in a hop-by-hop manner based on a routing tree. Message streams merge progressively as they approach the sink. As before, let us assume for the sake of discussion that the senders in the network generate Poisson flows, then by the superposition property of Poisson processes the combined stream arriving at node i of m independent Poisson processes with rate λ_j^i is a Poisson process with rate $\lambda^i = \lambda_1^i + \lambda_2^i + \cdots + \lambda_m^i$, where m is the number of "routing" children for node i. Additionally, we let $1/\mu_i$ be the average buffer delay injected by node i. Then node i is an $M/M/\infty$ queue, with arrival parameter λ^i and departure parameter μ_i, yielding:

- $N_i(t)$, the number of packets in the buffer at node i, is Poisson distributed.
- $p_{ik} = P\{N_i(t) = k\} = \frac{\rho_i^k}{k!}e^{-\rho_i}$, where $\rho_i = \lambda^i/\mu_i$.
- Expected number of messages at node i, $\overline{N_i} = \rho_i$.

As expected, if we choose our delay strategy at node i such that μ_i is much smaller than λ^i (as is desirable for enhanced temporal privacy), then the expected buffer occupancy $\overline{N_i}$ will be large. Thus, temporal privacy and buffer utilization are conflicting system objectives.

The last issue that we need to consider is the amount of storage available for buffering at each sensor. As sensors are resource-constrained devices, it is more accurate to replace the $M/M/\infty$ queues with $M/M/k/k$ queues, where memory limitations imply that there are at most k servers/buffer slots, and each buffer slot is able to handle one message. If an arriving packet finds all k buffer slots full, then either the packet is dropped or, as we shall describe later in Section 5, a preemption strategy can be employed. For now, we just consider packet dropping. We note that packet dropping at a single node causes the outgoing process to lose its Poisson characteristics. However, we further note that by Kleinrock's Independence approximation (the merging of several packet streams has an affect akin to restoring the independence of interarrival times) [4], we may continue to approximate the incoming process at node i as a Poisson process with aggregate rate λ^i. Hence, in the same way as we used a tree of $M/M/\infty$ queues to model the network earlier, we can instead model the network as a tree of $M/M/k/k$ queues.

The $M/M/k/k$ formulation provides us with a means to adaptively design the buffering strategy at each node. If we suppose that the aggregate traffic levels arriving at a sensor node is λ, then the packet drop rate (the probability that a new packet finds all k buffer slots full) is given by the well-known Erlang Loss formula for $M/M/k/k$ queues:

$$\alpha = E(\rho, k) = \frac{\rho^k}{k!}p_0 = \frac{\frac{\rho^k}{k!}}{\sum_{i=0}^{k} \frac{\rho^i}{i!}}, \tag{5}$$

where $\rho = \lambda/\mu$. For an incoming traffic rate λ, we may use the Erlang Loss formula to appropriately select μ so as to have a target packet drop rate α when using buffering to enhance privacy. This observation is powerful as it allows us to adjust the buffer delay parameter μ at different locations in the sensor network, while maintaining a desired buffer performance. In particular, the expression for $E(\rho, k)$ implies that, as we approach the sink and the traffic rate λ increases, we must decrease the average delay time $1/\mu$ in order to maintain $E(\rho, k)$ at a target packet drop rate α.

5. RCAD: Rate-Controlled Adaptive Delaying

As shown in the previous section, introducing delays prior to forwarding packets imposes buffer demands on intermediate nodes. Hence we need to adjust the delay distribution as a function of the incoming traffic rate and the available buffer space.

We propose *RCAD*, a Rate-Controlled Adaptive Delaying mechanism, to achieve privacy and desirable buffer performance simultaneously. The main idea behind RCAD is buffer preemption– if the buffer is full, a node should select an appropriate buffered packet, called the *victim packet*, and transmit it immediately rather than drop packets. Consequently, preemption automatically adjusts the effective μ based on buffer state. The victim packet is the packet that has the shortest remaining delay time. In this way, the resulting delay times for that node are the closest to the original distribution. Besides, the implementation of this

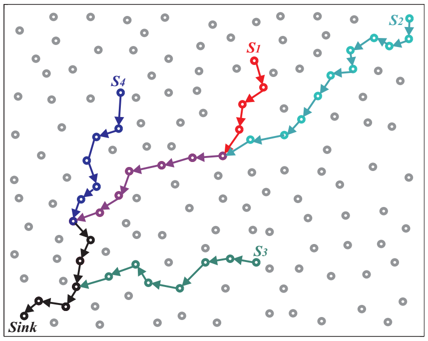
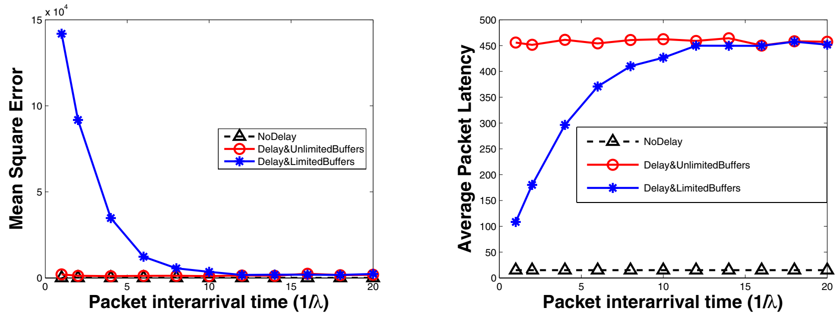


Figure 1. Simulation topology



(a) Mean square error

(b) Delivery latency

Figure 2. Temporal privacy in 1)no delay, 2) delay with unlimited buffers and 3) delay with limited buffers (RCAD)

scheme is straightforward because each node already keeps track of the remaining buffer time for every packet. In this study, we have developed a detailed event-driven simulator to study the performance of RCAD. We have set the simulation parameters, such as the buffer size and the traffic pattern, following measurements from an actual sensor platform, i.e. Berkeley motes, to model realistic network/traffic settings. Finally, we have measured important performance and privacy metrics.

5.1. Privacy and Performance Metrics

In our simulation studies, we have measured both the temporal privacy and network performance of RCAD. As discussed in Section 2.1, we use the *MSE* to quantify the error the adversary has in estimating the creation times of each packet. After introducing delays at intermediate nodes, we need to modify our adversary estimation model to accommodate this additional delay process. In addition to the knowledge the adversary has in Section 2.1, i.e. hop-count to the source and the average transmission delay at each node, we also assume the adversary knows the delay process for each flow, i.e. the delay distribution. Consequently, for an observed packet arrival time z, our adversary now estimates the creation time of this packet as $x' = z - y$, where y includes both the transmission delay and the additional delay. Here, our baseline adversary model uses the original delay distribution to calculate his estimations, neglecting the fact that some packets may have shorter delays than specified by the original delay distributions due to packet preemptions. Hence, our adversary estimates the delay for flow i as h_i/μ, where h_i is the hop count of flow i and $1/\mu$ is the average per hop delay. For a sequence of packets coming from one source, we use the MSE to measure the temporal privacy of the underlying network. As noted earlier, there is a direct relationship between the information theoretic metric (mutual information) of privacy we defined in Section 3 and mean square error [10]. The scheme that has a higher estimation error consequently better preserves the temporal privacy of the source.

Additionally, we note that it is desirable to achieve privacy while maintaining tolerable end-to-end delivery latency for each packet. Our objective is to introduce minimal extra latency while maximizing temporal privacy and, hence, we also examine the average latency induced by the RCAD algorithm.

5.2. Simulation Setup

The topology that we considered in our simulations is illustrated in Figure 1. Here, nodes S_1, S_2, S_3, and S_4 are source nodes and create packets that are destined for the sink. Thus, we had four flows, and these flows had hop counts 15, 22, 9 and 11 respectively. Each source generated a total of 1000 packets at *periodic intervals* with an inter-arrival time of $1/\lambda$ time units. In our experiments we varied $1/\lambda$ from 2 (i.e. the highest traffic rate) time units to 20 (the slowest traffic rate) to generate different cases of traffic loads for the network. The main focus of our simulator is the scale of the network, so we simplified the PHY- and MAC-level protocols by adopting a constant transmission delay (i.e. 1 time unit) from any node to its neighbors. Unlike the poisson traffic assumption used in Section 4, we use a realistic sensor traffic model where packets are periodically transmitted by each source. When a packet arrives at an intermediate node, the intermediate node introduces a random delay following an exponential distribution with mean $1/\mu$. Unless mentioned otherwise we took $1/\mu = 30$ time units in the simulations. The results reported are for the flow S_1 to the sink.

5.3. Effectiveness of RCAD

To study the effectiveness of the RCAD strategy, we compare temporal privacy in the following situations:

1. Nodes in the network forward packets as soon as they receive them. This scenario is the baseline case with no effort made to explicitly provide any temporal privacy.

2. Each node in the path of a network packet introduces an exponential delay with mean $1/\mu = 30$, before forwarding the packet. This scenario adds uncertainty

to the adversary's inference of the time of origin of a packet. In this case, we assume that the nodes have unlimited buffers.

3. Same as above except each node now has limited buffers. Specifically, we assume each node can buffer 10 packets, which approximates the buffers available on the Mica-2 motes. This models the real-world scenario where sensor nodes are resource constrained.

We used the topology in Figure 1 with 4 different traffic flows. Figure 2(a) shows the MSE in the adversary estimate for the 3 situations above, with regards to flow from S1. We can see that the error is very small in both cases 1 and 2 (it may appear to be zero, but that's only because of the relative scale as compared to case 3). For case 2, the MSE is small because the adversary has adjusted for the delay based on his knowledge of the delay distributions used. In case 3, however, the adversary, attempts to use his knowledge of the intermediate delay distributions (specifically the knowledge of $1/\mu$) to estimate the time of origin of packets. But the preemptions at higher traffic rates (small inter-arrival times), cause the *effective* latencies of the packets to be much lower than the *expected* latencies and this results in very high error in the adversary's estimate. Figure 2(b) shows the average latency for packets to reach from the source to the sink. As expected, case 1 has the lowest latency, as no artificial delays have been introduced. Note that case 2 shows the highest latency, which is the average of the combined delay distribution of all the nodes in the path of flow from S1. Further, in case 3, we find that the preemptions due to limited buffers actually help reduce the average delivery latency, especially for high source traffic rates (smaller inter-arrival times). For example at $1/\lambda = 2$, case 3 reduces the average latency by a factor of 2.5. These results clearly demonstrate the efficacy of RCAD algorithm in providing temporal privacy (high MSE for case 3) with controlled overhead in terms of average packet latency.

5.4. The Adaptive Adversary Model

Since RCAD scheme dynamically adapts the delay processes by adopting a buffer preemption strategy, it is inadequate for the adversary to estimate the actual delay times using the original delay distributions before preemption. Hence, we also enhance the baseline adversary to let the adversary adapt his estimation of the delays depending on the observed rate of incoming traffic at the sink. We call such an adversary as an *adaptive* adversary.

In order to understand our adaptive adversary model, let us first look at a simple example. Let us assume there is only one node with one buffer slot between the source and sink. Further, assume that the packet arrival follows a Poisson process with rate λ, and the buffer generates a random delay time that follows an exponential distribution with mean $1/\mu$. If the buffer at the intermediate node is full when a new packet arrives, the currently buffered packet will be transmitted. In this example, if the traffic rate is low, say $\lambda < \mu$, then the packet delay time will be $1/\mu$. However, as the traffic increases, the average delay time will become $1/\lambda$ due to buffer preemptions. Following this example, our adaptive adversary should adopt a similar estimation strategy: at low traffic rates, he estimates the overall average delay y by h/μ, while at higher traffic rates, he estimates the overall average delay y as a function of the buffer space and the incoming rate, i.e. hk/λ, where h is the flow hop count, k is the number of buffer slots at each node, and λ is the traffic rate of that flow. Given an aggregated traffic rate λ_{tot} from n sources converging at least one-hop prior to the sink, the adversary can compute the probability of buffer overflow via the Erlang Loss formula in equation (5). He can then compare this against a chosen threshold and if the probability is less than the threshold, he will assume the average delay introduced by each hop is $1/\mu$. However, if the probability is higher than the threshold, the average delay at each node is calculated to be nk/λ_{tot}.

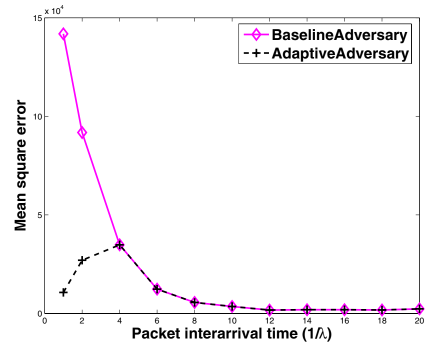


Figure 3. The estimation error for the two adversary models.

We studied the ability of an adaptive adversary to estimate the time of creation when using RCAD with identical delay distributions across the network. The resulting estimation mean square errors are presented in Figure 3. The adaptive adversary adopts the same estimation strategy as the baseline adversary at lower traffic rates, i.e. h_i/μ for flow i, but it uses the incoming traffic rate to estimate the delay at higher traffic rates, i.e. $h_i k/\lambda_i$ for the average delay of flow i. To switch between estimation strategies, the adversary used the Erlang Loss formula for a threshold preemption rate of 0.1. Figure 3 shows that the adaptive adversary can significantly reduce (but not eliminate) the estimation errors, especially at higher traffic rates (lower inter-arrival times) where preemption is more likely.

6. Related Work

The problem of privacy preservation has been considered in the context of data mining and databases [2, 13]. A common technique is to perturb the data and to reconstruct distributions at an aggregate level. A distribution reconstruction algorithm utilizing the Expectation Maximization (EM) algorithm is discussed in [1], and the authors showed that it converges to the maximum likelihood estimate of the original distribution based on the perturbed data.

Contextual privacy issues have been examined in general networks, particularly through the methods of anonymous communications. Chaum proposed a model to provide anonymity against an adversary conducting traffic analysis [5]. His solution employs a series of intermediate systems called mixes. Each mix accepts fixed length messages from multiple sources and performs one or more transformations on them, before forwarding them in a random order. Most of the early mix related research was done on *pool mixes* [9], which wait until a certain threshold number of packets arrive before taking any mixing action. Kesdogan [12] proposed a new type of mix, *SG-Mix*, which delays an individual incoming message according to an exponential distribution before forwarding them on. Later, Danezis proved in [7] using information theory that a SG-Mix is the optimal mix strategy that maximizes anonymity. The objective of SG-Mixes, however, is to decorrelate the input-output traffic relationships at an individual node, and the methods employed do not extend to networks of queues.

Source location privacy in sensor networks is studied in [11, 14], where phantom routing, which uses a random walk before commencing with regular flooding/single-path routing, protects the source location. In [8], Deng proposed randomized routing algorithms and fake message injection to prevent an adversary from locating the network sink based on the observed traffic patterns.

7. Concluding Remarks

Protecting temporal context of a sensor reading in a sensor network cannot be accomplished by merely using cryptographic mechanisms. In this paper, we have proposed a technique complimentary to conventional security techniques that involves the introduction of additional delay in the store-and-forward buffers within the sensor network. We formulated the objective of temporal privacy using an information-theoretic framework, and then examined the effect that additional delay has on buffer occupancy within the sensor network. Temporal privacy and buffer utilization were shown to be objectives that conflict, and to effectively manage the tradeoffs between these design objectives, we proposed an adaptive buffering algorithm, RCAD (Rate-Controlled Adaptive Delaying) that preemptively releases packets under buffer saturation. We then evaluated RCAD using an event-driven simulation study for a large-scale sensor network. We observed that, when compared with a baseline network with no artificially introduced delays, RCAD was able to provide enhanced temporal privacy with a controlled latency overhead. We further showed that RCAD was able to sustain a better performance than baseline case even with an improved adversary model.

References

[1] D. Agrawal and C. C. Aggarwal. On the design and quantification of privacy preserving data mining algorithms. In *Symposium on Principles of Database Systems*, 2001.

[2] R. Agrawal and R. Srikant. Privacy-preserving data mining. In *Proc. of the ACM SIGMOD Conference on Management of Data*, pages 439–450. ACM Press, May 2000.

[3] V. Anantharam and S. Verdu. Bits through queues. *IEEE Trans. on Information Theory*, 42:4–18, 1996.

[4] D. Bertsekas and R. Gallager. *Data Networks*. Prentice Hall, 1992.

[5] D. Chaum. Untraceable electronic mail, return addresses, and digital pseudonyms. *Communications of the ACM*, 24:84–88, 1981.

[6] T. Cover and J. Thomas. *Elements of Information Theory*. John Wiley and Sons, 1991.

[7] G. Danezis. The traffic analysis of continuous-time mixes. In D. Martin and A. Serjantov, editors, *Privacy Enhancing Technologies (PET 2004)*, May 2004.

[8] J. Deng, R. Han, and S. Mishra. Countermeasures against traffic analysis attacks in wireless sensor networks. In *First IEEE/CreateNet Conference on Security and Privacy for Emerging Areas in Communication Networks*, 2005.

[9] C. Diaz and B. Preneel. Taxonomy of mixes and dummy traffic. In *3rd Working Conference on Privacy and Anonymity in Networked and Distributed Systems*, 2004.

[10] D. Guo, S. Shamai, and S. Verdu. Mutual information and minimum mean-square error in gaussian channels. *IEEE Trans. on Information Theory*, 51(4):1261–1282, 2005.

[11] P. Kamat, Y. Zhang, W. Trappe, and C. Ozturk. Enhancing source-location privacy in sensor network routing. In *ICDCS '05: Proceedings of the 25th IEEE International Conference on Distributed Computing Systems (ICDCS'05)*, 2005.

[12] D. Kesdogan, J. Egner, and R. Buschkes. Stop-and-go-mixes providing probabilistic anonymity in an open system. In *Proceedings of the Second International Workshop on Information Hiding*, pages 83–98, 1998.

[13] C. K. Liew, U. J. Choi, and C. J. Liew. A data distortion by probability distribution. *ACM Trans. Database Syst.*, 10(3):395–411, 1985.

[14] C. Ozturk, Y. Zhang, and W. Trappe. Source-location privacy in energy-constrained sensor network routing. In *SASN '04: Proceedings of the 2nd ACM workshop on Security of ad hoc and sensor networks*, 2004.

[15] A. Perrig, R. Szewczyk, D. Tygar, V. Wen, and D. Culler. SPINS: security protocols for sensor networks. *Wireless Networks*, 8(5):521–534, 2002.

[16] R. Szewczyk, A. Mainwaring, J. Polastre, J. Anderson, and D. Culler. An analysis of a large scale habitat monitoring application. In *SenSys '04: Proceedings of the 2nd international conference on Embedded networked sensor systems*, pages 214–226. ACM Press, 2004.

[17] W. Trappe and L. Washington. *Introduction to Cryptography with Coding Theory*. Prentice Hall, 2002.

[18] A. Woo, T. Tong, and D. Culler. Taming the underlying challenges of reliable multihop routing in sensor networks. In *Proceedings of the 1st international conference on Embedded networked sensor systems*, pages 14–27, 2003.