

Battle Space Sensornets made Intelligent against Adversaries

Ujwal K. Chinthala, Abhishek Ray Chaudhuri and Amiya Bhattacharya
New Mexico State University, Las Cruces, NM 88003-8001
Email: {ujwal_27, abhishek, amiya}@nmsu.edu

December 26, 2006

Abstract

Sensornets are being popularly deployed within a battle field for exchanging information between sensitive hideouts of soldiers and their ammunition camps. Battlefields are spread over a vast expanse and thus cannot be physically secured from eavesdropping intruders. Under these circumstances, the origin of sensed events becomes important as the wireless hop by hop route taken by these messages leave a trail to the source, for adversaries. Enemies can use traffic analysis attacks to localize the source and thereby undermine the security of the battle space. The context of the sensed information can be protected by using spatial obfuscation techniques. However, these techniques are communication intensive and results in a higher drainage of battery power. Our complementary techniques involve lazy-reporting (coalescing) of the source data prior to encryption. Data coalescing not only hides the temporal pattern of the source events, but also saves valuable battery power by reducing wireless transmissions. We simulate these obfuscation techniques against two adversarial scenarios. Adversaries can be defined as mobile sensor nodes with compatible hardware to any other sensor node in sensornet. The adversary tries to localize the source and time of the occurrence of event. We contemplate adversarial network scenario that is more powerful and practical in a battle space. We create a network of adversaries that can conspire to localize the source.

We perform an extensive protocol simulation using an event driven simulator. The main challenges in this simulation were to implement a non-traditional phantom flooding used for preserving contextual privacy in sensornets and create an intelligent adversarial protocol which reacts to the traffic in the network. The adversarial network is simulated as virtual network embedded into the sensornet which does not interfere with the operation of sensornet but converges towards the source by conspiring with each other.

Simulation experiments have been performed over the existing spatial obfuscating techniques. The results show good performance of these techniques against a single adversary,

although at the expense of increased battery power. Our results show that spatial obfuscation techniques when complemented with coalescing techniques decrease the energy requirements by more than half. We also show that the spatial obfuscating techniques which perform well against a single adversary, gives suboptimal performance against powerful adversaries. But when these spatial obfuscating techniques are complemented with coalescing techniques the safety period of the source until capture, improves considerably.

1 INTRODUCTION

Sensornets are used in many applications like, monitoring battlefield, habitat monitoring and moving vehicle tracking. There has been an increasingly wide deployment of sensor networks for all/some of the above applications. Privacy in sensor networks can be categorized into two classes: content-oriented privacy and contextual privacy [5]. Content-oriented privacy is the ability of adversaries to learn the content of transmissions in sensor networks [5]. Contextual privacy considers the ability of adversaries to infer information from observations of sensors and communications without access to the content of messages. Traffic analysis attacks described in [3], infer us to the capability of the adversaries, who can observe communication patterns and a messages routing path deduce information about the network, such as the location of a messages source or the location of the base station. Contextual Privacy is the main concern of deployment of sensornets in wide area deployments like battlefields and habitat monitoring, as these areas cannot be physically secured from intruders [7]. This paper focuses on the problem of protecting the location of a source in sensor networks. The goal of our research is to increase the time for an adversary to find a source, which is called the safety period [5] and also reduce the capture probability [5] i.e, minimize the chance of an adversary finding the source.

Contextual Privacy in sensornets becomes a very important problem when a sensor-net is used in monitoring valuable/sensitive assets. For example, on a battlefield sensornet the movements of soldiers are detected by sensors and are reported to the headquarters. An adversary can then intercept sensornet communications and determine the exact location of opposing soldiers through traffic analysis. There have been couple of advanced routing algorithms like phantom flooding [5], CEM [17] and fractal propagation [3] for the contextual privacy protection in sensornets. But these methods, which are spatial obfuscation techniques are communication intensive and hence, provide contextual privacy at the extent of high energy requirements.

Adversaries used in [5], [17] and [3] deal with adversaries mote-class attackers [17]. Mote-class attackers have capabilities similar to those of a sensor node. The radius they can eavesdrop may be the same as, or perhaps up to twice as great as, a sensor nodes communication

radius. At any given time a mote-class attacker can thus only detect communication between a limited number of nodes. In this paper, we contemplate a more powerful adversary called adversarial network, which is quite feasible in a battle field or habitat monitoring environment. An Adversarial Network is a network of multiple mote-class adversaries, which can co-operate and exchange network information among themselves out-of-band to localize the source.

There has been some research on the source location protection problem. Phantom flooding technique discussed in [5] uses random walk to create phantom sources all around the sensor network, to attract the adversaries towards these phantom sources. CEM [17] uses loops to increase the safety period by dragging the adversaries into the loops which transmit fake messages. In this paper, we propose coalescing techniques which can be used as add-on feature to spatial coalescing techniques, to enhance the contextual privacy by lazy-reporting the data, at the same time reducing the energy requirements to half. We also investigate, how power of adversary affects the performance of routing algorithms which provide contextual privacy and those embedded with coalescing techniques. The rest of this paper is organized as follows. Section 2 discusses related work in more detail. Section 3 describes the adversary/adversarial network concept. Section 4 discusses temporal obfuscation techniques and its performance. Section 5 discusses the simulation environment used to show the effectiveness of temporal obfuscation techniques. Section 6 describes experiments and results applying temporal obfuscation techniques and analyzes its performance. Section 7 concludes this paper.

2 RELATED WORK

Contextual privacy is an integral part of total privacy protection in a sensor network environment and needs to be handled with utmost importance. Nevertheless, this problem remains relatively new in the field of sensor networks, since very feeble efforts have gone into developing new protocols or bolstering existing ones to support contextual privacy. Researchers are increasingly raising concerns about privacy issues in sensor networks. Ozturk et al. [9] introduced the Panda-Hunter model to formalize the source location problem in sensor networks and also proposed the phantom flooding approach. Kamat et al. [5] extended the work of [9] and proposed phantom routing techniques based on both flooding and single-path routing. Although the techniques discussed in [5] perform badly with single-path routing. In paper [17] the authors try to increase the safety period of single path routing by creating loops on the routing path, which produce fake messaging. Deng et al. [3] investigated several countermeasures against traffic analysis that aimed to protect the location of a base station. Chaums mixing approach [1] was shown to provide anonymous connections that protected against traffic analysis and were useful in an

onion routing mechanism [14]. Several other secure routing protocols have been proposed, such as [12], [13], and [11].

3 ADVERSARY MADE POWERFUL

Adversaries can be defined as mobile sensor nodes with compatible hardware to any other sensor node in sensornet. The adversary tries to localize the source and time of the occurrence of event. Adversaries eavesdrop on the traffic between sensor nodes within its radio range to determine the source nodes location. We assume that all the messages are encrypted with secret keys, and thus adversaries cannot determine the content of messages even if they intercept or eavesdrop on communications. However, an attacker may use RF localization techniques to trace back hop-by-hop to a sources location. We also assume that an adversary can determine the sender of each message it intercepts, and the adversaries use the patient adversary model, as described in [9] and shown in Adversary model 1. Moreover the adversary is assumed to be non-malicious, device rich resource rich and informed as described in [9]. We contemplate two types of adversaries that are practical in a sensornet. These different types of adversaries are discussed below in detail.

3.1 ADVERSARIAL MODEL 1

Adversary in this model has capabilities similar to those of a sensor node. The radius an adversary can eavesdrop will be the same as, a sensor nodes communication radius. At any given time an adversary can only detect communication between a limited number of nodes. In this model we use patient adversary model, A patient adversary always selects the latest messages sender *sender_message* as the next node to move to. Assuming a sensor node will not forward the same message twice and that the sensor network is using broadcast routing, *sender_message* should be the nearest node to the sources location among all the neighbors of the current node. Thus, under the assumptions of this model, moving to *sender_message* every time one detects a message will move closer and closer to the source, eventually reaching it. The patient adversary model is discussed in [9] and the pseudo-code is given below:

3.2 ADVERSARY MODEL 2 (ADVERSARIAL NETWORK)

Adversarial Network is a network of adversaries whose hardware is compatible to the sensornodes in sensornet. The eavesdropping radius of each adversary is same as, a sensor node's communication radius. But eavesdropping area of the overall adversarial network is much more than, a sensor node's communication range. Thus at any time, an adversary the Adversarial

Algorithm 1 Adversary Model 1

```
1: adversary radio range = sensor node radio range
2: Start from the sink
3: while did not find source location do
4:   Eavesdrops on the communications within its radio range
5:   Receive a new message new_message
6:   Determine the sender sender_new_message of new_message
7:   Move to the new location sender_new_message
8: end while
```

Network Model 1 can detect communication between a vast number of nodes and also these adversaries use out-of-band communication to communicate with each other and conspire to localize the source. The pseudo-code for this model is given below:

Algorithm 2 Adversarial Network

```
1: adversary radio range = sensor node radio range
2: waiting_time = mean inter arrival time of events
3: Start from the sink
4: while did not find source location do
5:   Eavesdrops on the communications within its radio range
6:   if did not receive message for more than waiting_time then
7:     for each adversarial peer do
8:       Ask their recent sender's (sender_info)
9:       if sender_info is stale then
10:        Don't use the sender_info
11:       else
12:         adversary_confidence = adversary_confidence + 1
13:         calculate the sum of the adversary peers sender_info
14:       end if
15:     end for
16:     if total number of adversaries is 2 then
17:       move one hop distance towards the sender_info of the other adversary
18:     else
19:       calculate the mean of the adversaries sender_info
20:       mean of X-Coordinate's and Y-Coordinates of sender_info (new_location)
21:       move one hop distance towards the new_location
22:     end if
23:   else
24:     Determine the sender sender_new_message of new_message
25:     Move to the new location sender_new_message
26:   end if
27: end while
```

Whenever an adversary intrudes a sensornet with the intention of tracking down the context of triggering events, it is expected to be sufficiently equipped with devices such as antenna and spectrum analyzers. With the help of these devices, he can measure the angle of arrival of a message and the received signal strength to make a movement towards the immediate

sender. The adversary can also move at any rate and has an unlimited supply of power to sustain the long-term operation of its sophisticated devices. In addition, he is provided with a large amount of memory or storage devices to keep track of messages that have been heard by him and record the nodes he has visited so far. Thus, the device-rich and resource-rich nature of the adversaries place them in an advantageous position as compared to the resource-poor sensor nodes.

Moreover, The spatial obfuscation techniques like Phantom flooding, fractal propagation, CEM, try to confuse the adversary by creating fake sources. These schemes provide contextual privacy by moving single adversary towards the phantom sources, instead of the actual source. These schemes act powerful against single adversary as he does not have the global view of the traffic in the sensornet. Although, there performance deteriorates when used against a adversarial network, since in an adversarial network we spread the adversarial nodes in the sensornet so that at any time the adversaries can gather more information about the activity in the sensornet. Also, the adversaries share the information among themselves, which helps each adversary to have a global view of the activity in the sensornet.

4 TEMPORAL OBFUSCATION

4.1 A renewal model for coalescing

Sensor readings at its source can be abstracted as a time series data along with appropriate timestamps. Starting from time $t = 0$, let the sequence of timestamps $\mathbf{S} = \{S_i, i \geq 1\}$ denote the time when an event triggers causing the i -th sensor reading to be recorded. Let the sequence $\mathbf{T} = \{T_i, i \geq 1\}$ be the inter-arrival time between these events. Without loss of generality, let the inter-arrival times $\{T_i\}$ be positive, independent and identically distributed (iid) with a common distribution $F(t)$ such that $F(0) < 1$. The related counting process $\{C(t), t \geq 0\}$, the number of sensing events by time t , is a renewal process. The key idea behind the renewal modeling of the sensing events is that the process probabilistically starts over at every renewal (event), due to the iid inter-arrival times. The temporal context of the sensed data can be defined as the realization of the sequence $\{S_i\}$. The prime objective of temporal obfuscation would be to increase the uncertainty of the stochastic process S . *Shannon's entropy* [2] is used to quantify this uncertainty.

For brevity, let us introduce the notation S^n for the finite sequence S_1, \dots, S_n , with S^0 denoting the sequence with no elements. Similar notation T^n applies to the sequence T_1, \dots, T_n . Thus, for a coalesced sequence of n sensor readings, $H(S^n)$ and $H(T^n)$ represent the uncertainty

or descriptive complexity involved in the timestamps and inter-arrival times respectively. Because of the bijection between the finite sequences S^n and T^n , $H(S^n)$ and $H(T^n)$ are equivalent.

This leads to a simple protocol for sensor data coalescing where sensor readings are accumulated locally until every n event triggers. The parameter n is the only source of tunability and needs to be chosen appropriately based on how responsive the sensing application needs to be. Simple coalescing introduces only a limited amount of uncertainty to $H(T^n)$ and becomes trivial for the adversary to guess the event timestamps if it is known that simple coalescing is taking place with a known n . Even if n is unknown, it is easy for the adversary to guess it using a brute force trial.

4.2 Coalescing as a randomly stopped sequence

The simple coalescing algorithm can be modified by choosing an appropriate probability distribution for a random variable N so that it is a *stopping time* for the sequence $\{T_i\}$. The realization of the stopping time N is used instead of fixed n as a threshold for sending out the coalesced data sequence.

The timestamps of the coalesced data sequence can thus be represented by the randomly stopped sequence $T^N \in \mathcal{T}^*$, where \mathcal{T}^* denotes the set of all finite length sequences created with support set \mathcal{T} . The uncertainty involved in the timestamps of these coalesced sequence needs to be investigated. To this goal, we need to analyze the entropy $H(T^N)$ of the sequence $T^N = T_1, \dots, T_N$. A theorem in [4] that proves a Wald-like equation for the entropy of a stopped sequence is used.

Result 1 *Let N be a stopping time for the sequence of iid random variables T_1, T_2, \dots . Then*

$$H(T^N) = E[N]H(T_1) + H(N|T^\infty) \quad (1)$$

4.3 A taxonomy of coalescing algorithms

For N to be a stopping time for the sequence $\{T_i\}$, the event $\{N = n\}$ must be independent of T_{n+1}, T_{n+2}, \dots . Following are the two natural choices of how the event $\{N = n\}$ can depend on $T^n = \{T_1, \dots, T_n\}$.

4.3.1 Independent coalescing

Given that N is chosen as a random variable independent of $\{T_i\} = T^\infty$, setting $H(N|T^\infty) = H(N)$ in equation (1)

$$H(T^N) = E[N]H(T_1) + H(N). \quad (2)$$

It is apparent from equation (2) that one must maximize $E[N]$ to maximize $H(T^N)$, subject to that constraint that $Pr[\sum_{i=1}^N T_i \geq \delta] \leq \epsilon$ for some intended choice of deadline δ and the probability ϵ of missing it. Using Markov inequality along with *Wald's equation* [?], we obtain

$$Pr \left[\sum_{i=1}^N T_i \geq \delta \right] \leq (E[N]E[T_1])/\delta \quad (3)$$

Setting this upper bound to ϵ , an upper bound on $E[N]$ can be established as $E[N] \leq \delta\epsilon/E[T_1]$.

Once $E[N]$ is maximized, we can use the principle of entropy maximization to select the *maximum entropy probability distribution (MEPD)* subject to our choice of its mean. Thus, N must be geometrically distributed with per slot success probability $E[T_1]/(\delta\epsilon)$ [6].

4.3.2 Deterministic coalescing

The choice of $\{N = n\}$ can be determined completely by $\{T_1, \dots, T_n\}$ by choosing a suitable threshold τ such that we stop after the first sensing event after τ . According to notation of our renewal model, this is the instant of $(C(\tau) + 1)$ -th renewal. Given that N is completely determined by T^N , we can set $H(N|T^\infty) = 0$ in equation (1) to obtain

$$H(T^N) = E[N]H(T_1). \quad (4)$$

For our particular choice of $N \equiv C(\tau) + 1$, this reduces to

$$H(T^N) = (m(\tau) + 1)H(T_1). \quad (5)$$

To maximize $H(T^N)$, we need to maximize $m(\tau)$ subject to the constraint $Pr[\sum_{i=1}^{C(\tau)+1} T_i \geq \delta] \leq \epsilon$ as in the previous case. Once again, Markov inequality gives

$$Pr \left[\sum_{i=1}^{C(\tau)+1} T_i \geq \delta \right] \leq (m(\tau) + 1)E[T_1]/\delta \quad (6)$$

Setting this upper bound to ϵ , we see that the threshold τ should be chosen such that $m(\tau)$ has the upper bound $\delta\epsilon/E(T_1) - 1$.

5 SIMULATION ENVIRONMENT

This chapter presents the simulation environment which was used to test the effectiveness of 1) Coalescing techniques, for enhancing privacy in wireless sensor networks 2) Privacy enhanced routing algorithms against Adversarial network and 3) The effects of power of adversary on 1) and 2). The goal of the simulation process is to verify the correct operation of our coalescing algorithms and evaluate its performance using simulation, and also to evaluate the performance of our coalescing algorithms against adversarial network. We validate our work by comparing it with one of the current state-of-the art protocols present for contextual privacy in wireless sensor networks(Phantom Flooding).

5.1 NS2 Simulator

The performance evaluation of our algorithm is performed using the NS2 simulator [8]. NS2 is LBNL's Network Simulator. NS2 is extensively used by the networking research community. It provides substantial support for simulation of routing, multicast protocols over wired and wireless (local and satellite) networks, etc. It is discreet event-driven, object-oriented and runs in a non-realtime fashion. It consists of C++ core methods and uses Tcl and Object Tcl shell as interface allowing the input file(simulation script) to describe the model to simulate. Users can define arbitrary network topologies composed of nodes, routers, links and shared media. A rich set of protocol objects can then be attached to nodes, usually as agents. A user can modify existing protocol models or generate code from scratch for custom protocols and for reporting special statistics.

Agents represent endpoints where network-layer packets are constructed or consumed, and are used in the implementation of protocols at various layers. The class Agent has an implementation partly in OTcl and partly in C++. We created new routing agent(Phantom Flooding), adversary agent and wrote various Object Tcl scripts for coalescing techniques. We also modified various files of NS2 to integrate our agents into the NS2 environment. The design and implementation of Phantom Flooding Agent, Adversarial Network Agent and coalescing techniques are discussed below.

5.2 Design Steps for the Phantom Flooding Routing Protocol

The approach that was followed in developing the Phantom Flooding used throughout this research is as follows: We wrote a general overview of the phantom flooding routing algorithm, which offers better contextual privacy than any other spatial obfuscation techniques in

wireless sensor networks. We changed the random walk in phantom flooding to reflective random walk to increase its data delivery ratio. From the description of the proposed algorithms, we developed Phantom Flooding Routing Agent(C++ code) to determine the functionality of the Phantom Flooding. We added the Phantom Flooding Routing Agent to the NS2 environment. After compiling the NS2, Phantom Flooding was ready to be used as routing algorithm in the NS2 environment. Once the data has been gathered and examined, a statistical analysis of the data was determined using Matlab and Perl scripting.

5.3 Design Steps for the Proposed Coalescing Algorithm

Coalescing techniques are complimentary to any privacy aware routing algorithms. We added our coalescing techniques as a complementary feature to the Phantom Flooding routing protocol. The approach that was followed in developing the Coalescing Algorithms used throughout this research is as follows: We wrote a general overview of the proposed coalescing algorithms (i.e.,Independent and Simple Coalescing algorithms), which can be embedded into any spatial obfuscation techniques in wireless sensor networks. From the description of the proposed algorithms, we developed Object Tcl scripts for simple and independent coalescing algorithms, which can be used in NS2 environment.

5.4 Design Steps for the Proposed Adversarial Network

The approach that was followed in developing the proposed adversarial network throughout this research is as follows: We wrote a general overview of the proposed algorithm focusing on improving the strength and capabilities of adversary. From the description of the proposed algorithm we developed Adversarial Network Agent(C++ code) to determine the functionality of adversarial node in adversarial network. We added the Adversarial Network Agent to the NS2 environment. After compiling the NS2, the adversarial network was ready to be used against spatial obfuscation techniques(with and without coalescing). Once the data has been gathered and examined, a statistical analysis of the data was determined using Matlab and Perl scripting.

5.5 Phantom Flooding Agent, Adversarial Network Agent and NS2

We implemented Phantom Flooding in NS2 by creating four files phantomroutingagent.cc, phantomroutingagent.h, phantomroutingpacket.h and PRNeighbor.h which contain the functionality of Phantom Flooding. phantomroutingagent.h is the header file where we defined all necessary timers(Hello and Neighbor Timer) and routing agent which performs Phantom Flooding's functionality. In phantomroutingagent.cc we implemented all timers, routing agent

and Tcl hooks. In `phantomroutingpacket.h` we declared all packets used by Phantom Flooding needs to exchange among nodes in the sensornets. `PRNeighbor.h` is header file where we defined the neighbor class which is used for maintaining neighbor list for each node in sensornet.

Similarly, we implemented the Adversarial Network in NS2 by creating three files `PRAdversary.cc`, `PRAdversary.h`, `PRSource.h` which contain the functionality of Adversarial network. `PRAdversary.h` is the header file where we defined all necessary mobility methods and adversary agent which performs Adversary functionality. In `PRAdversary.cc` we implemented all mobility methods, adversary agent and Tcl hooks. We also made changes to files `common/packet.h`, `trace/cmu-trace.h,tcl/lib/ns-packet.tcl`, `tcl/lib/ns-default.tcl` and `tcl/lib/ns-lib.tcl` and `common/mobilenode.h` to integrate, assign textual name for our packet type, achieve trace support, give default values to binded attributes and to provide necessary infrastructure to create wireless nodes running Phantom Flooding and also to provide mobile infrastructure for adversaries in ns2 simulator.

5.6 Network Scenarios

The network to be simulated has three scenarios, scenario 1 has 490-580 sensor nodes(static) that have been randomly distributed on the sensor field within 1000 *meters* by 1000 *meters* grid. Each sensor node has a radio propagation range of 75 meters. The IEEE 802.15.4 was used as the medium access control protocol. A free space propagation model with threshold cutoff was used. The source node and sink node are predefined in this scenario. The source node senses the events, which are reported to the designated sink node. We wrote various simulation scripts to generate normal, geometric and uniform events at the source node.

Scenario 2 has 540 sensor nodes(static) that have been randomly distributed on the sensor field within 1000 *meters* by 1000 *meters* grid. Each sensor node has a radio propagation range of 75 meters. The IEEE 802.15.4 was used as the medium access control protocol. A free space propagation model with threshold cutoff was used. The Sink node is predefined in this scenario. The distance between the Source node and Sink is varied from \$300 - \$750 meters. The source node senses the events, which are reported to the designated sink node. We wrote various simulation scripts to generate normal, geometric and uniform events at the source node. We define an adversarial node which is placed at the sink. Adversary node has the same parameters as any sensor node in the sensor field. The adversary listens to the sensornet traffic and responds to it, by moving towards the source of the packet.

Scenario 3 has 540 sensor nodes(static) that have been randomly distributed on the sensor field within 1000 *meters* by 1000 *meters* grid. Each sensor node has a radio propagation

range of 75 meters. The IEEE 802.15.4 was used as the medium access control protocol. A free space propagation model with threshold cutoff was used. The Sink node is predefined in this scenario. The distance between the Source node and Sink is varied from 300 - 750 meters. The source node senses the events, which are reported to the designated sink node. We wrote various simulation scripts to generate normal, geometric and uniform events at the source node. We also define an adversarial network, one of the adversarial node is placed at the sink. The total number of adversaries in Adversarial network varies from 2-8. Adversarial nodes has the same parameters as any sensor node in the sensor field. The adversaries in the Adversarial network listens to the sensornet traffic and responds to it, by moving towards the source of the packet. The adversary who does not receive a packet for a Threshold time, collaborates with its peers in the adversarial network and tries to localize the source.

5.6.1 Assumptions Used in the Simulation Process

The following assumptions were considered when we did our experiments:

1. The packets are assumed to be encrypted, hence an adversary is not allowed to look into the contents of packet.
2. We use reflective random walk in the Phantom Flooding.
3. We use 0.8 as forwarding threshold of probabilistic flooding in Phantom Flooding.

6 RESULTS AND DISCUSSION

This paper evaluated the performance of coalescing techniques embedded into phantom flooding. We compared the performance of phantom flooding embedded with coalescing to phantom flooding(without coalescing). We also evaluated the performance of coalescing techniques against various powerful adversaries. Four metrics were chosen to evaluate the performance of the coalescing algorithms. The metrics were chosen to evaluate the efficiency in addition to the effectiveness of the coalescing algorithms. These metrics are: 1) Packet delivery ratio, the ratio of data packets delivered to the sink to those generated by source; 2) Mean number of packets per sensor node event, average number of packets(sent/recieved) per node per event; 3) Mean number of events before capture, Average number of events generated before adversary/adversarial network captures the source; 4) Probability of capture, the ratio of successful attempts made by adversary/adversarial network to capture the source by total number of attempts made by adversary/adversarial network.

Our experiments were done as a function of the following experimental factors: 1) Mean Neighbor density; 2) Compression ratio; 3) Distance between source and sink; 4) Radio range of adversary; 5) Number of adversaries

6.1 When to stop generating new data

Choose an acceptable value d for the standard deviation of the estimator. We chose 95% as value of d , for our experiments. Generate at least 20 data values. Continue to generate additional data values, stopping when you have generated k values and

$$S/\sqrt{k} < d, \tag{6.1}$$

where S is the sample standard deviation based on those k values. The estimate of θ is given by $\bar{X} = \sum_{i=1}^k X_i/k$ (7)

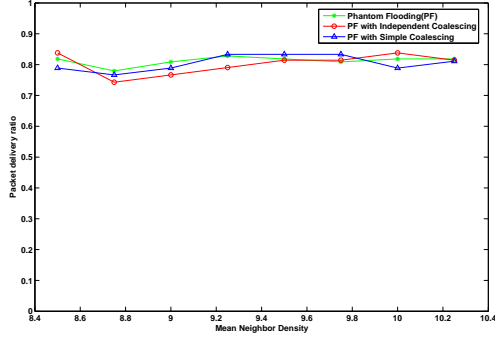
6.2 Experimental Factors

In our simulation scenarios, all nodes are distributed randomly in the sensor field; the density of these nodes depends on two factors, a parameter called seed in NS2 and the number of sensor nodes in the sensor field. We varied the density of sensor nodes by using constant seed through various experiments but with different number of nodes in the sensor field. In our experiments we use 95 percent confidence interval of the collected sample points for the tested protocols.

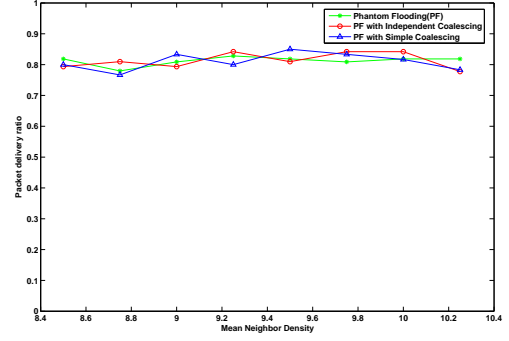
In this research, we examined the efficiency of coalescing techniques under a number of varying conditions through many experiments on the prescribed scenarios. Our simulation experiments were done by using different experimental factors. Those factors are: Neighbor Density (Nd): The neighbor density was varied from 8.0 to 10.0. Compression Ratio (Cr): In this compression ratio is varied in the range of (0.25, 0.5). Inter-arrival times (Ia)(normal, uniform, geometric): All the experiments, used to test the reliability of proposed algorithms are done with 3 different types of random events. Number of Adversaries (Na): The number of adversaries is varied in the range of (1, 2, 4, 8). Distance between Source and Sink (ssd): In this experiment we vary the source and sink distance in the range of (12, 18, 24, 30)hop distance.

6.3 SIMULATION RESULTS: SIMULATION SCENARIO 1

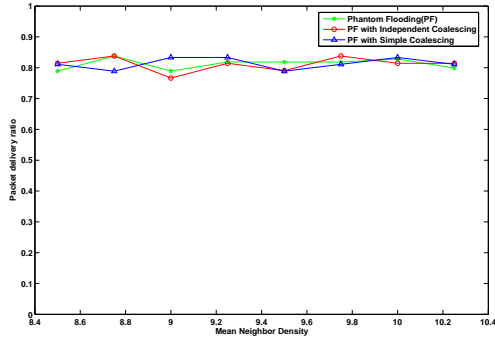
The main performance metric of interest in this research is the data delivery ratio. This metric will reflect the reliability of the correct operation of the phantom flooding we implemented, and also the reliability of our coalescing algorithms.



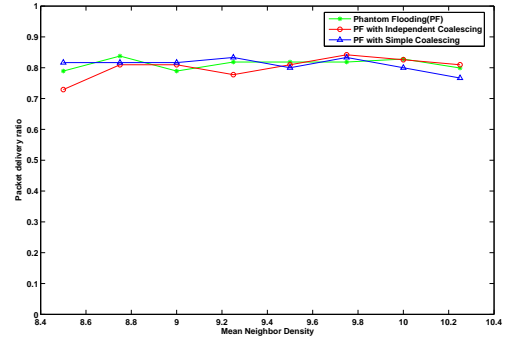
(a) Normally distributed events, compression ratio=0.25



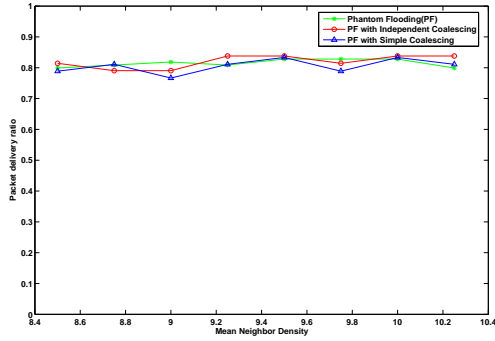
(b) Normally distributed events, compression ratio=0.5



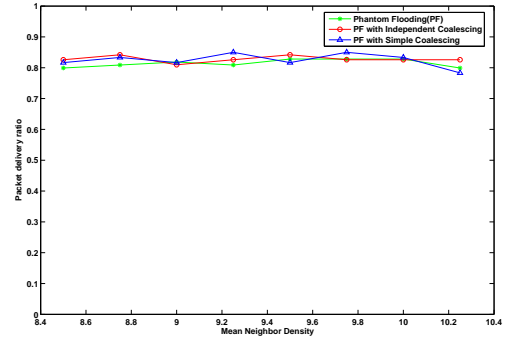
(c) Geometrically distributed events, compression ratio=0.25



(d) Geometrically distributed events, compression ratio=0.5



(e) Uniformly distributed events, compression ratio=0.25



(f) Uniformly distributed events, compression ratio=0.5

Figure 1: Packet delivery ratio as a function of Mean Neighbor density for normally, geometrically and uniformly distributed events, compression ratio = 0.25/0.5

As we see from the figure 1, The packet delivery ratio of phantom flooding lies between 75-85% for various neighbor densities. The probabilistic flooding with 80% threshold gives 90% packet delivery ratio in smaller networks. The mean packet delivery ratio from our results is 79%, since we are dealing with huge network with smaller transmission range. Moreover, this is the worst case packet delivery ratio in the sensornet as the source-sink distance we used in our experiments is the maximum possible distance between any two sensor nodes in sensornet.

Table 1: Data delivery ratio of PFIC and PSIC as %(data delivery ratio of PF), normally distributed events, compression ratio=0.25/0.5

	0.25	0.5
Phantom Flooding(PF)	100	100
Phantom Flooding with Independent Coalescing(PFIC)	98.5966	98.4888
Phantom Flooding with Simple Coalescing(PFSC)	96.3127	99.7246

Data delivery ratio of PFIC and PSIC as %(data delivery ratio of PF), geometrically distributed events, compression ratio=0.25/0.5

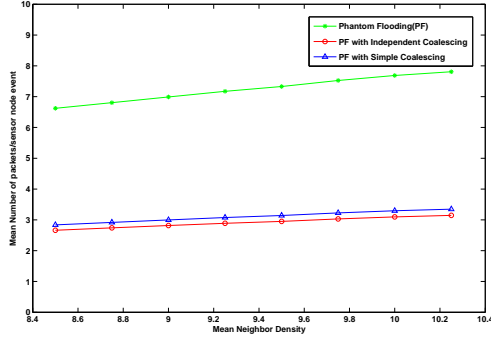
	0.25	0.5
Phantom Flooding	100	100
Phantom Flooding with Independent Coalescing	98.5966	98.5966
Phantom Flooding with Simple Coalescing	96.3127	96.3127

Table 2: Data delivery ratio of PFIC and PSIC as %(data delivery ratio of PF), uniformly distributed events, compression ratio=0.25/0.5

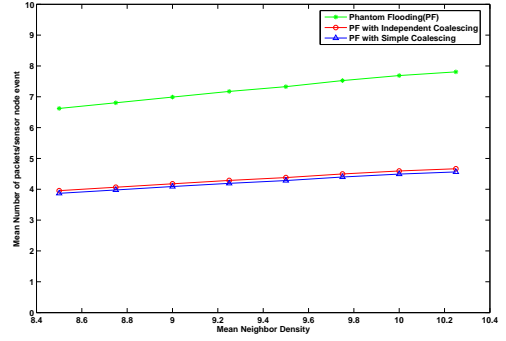
	0.25	0.5
Phantom Flooding	100	100
Phantom Flooding with Independent Coalescing	98.5966	102.1679
Phantom Flooding with Simple Coalescing	96.3127	101.7717

The table 1 thru table 2 gives the percentile decrement in packet delivery ratio of phantom flooding embedded with coalescing algorithms when compared to phantom flooding.

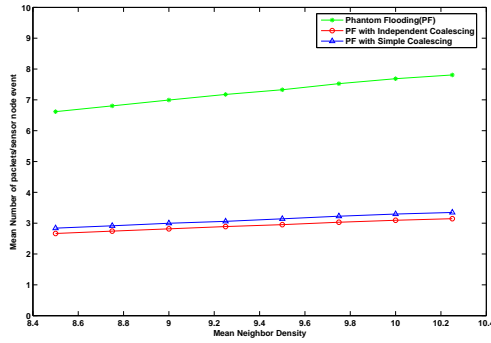
We can conclude from the tables that the independent coalescing deteriorates the data delivery ratio of phantom flooding by 1.4% and simple coalescing deteriorates the packet delivery ratio of phantom flooding by 3.7% for normally, geometrically and uniformly distributed events with 0.25 compression ratio. The packet delivery ratio is not effected by compression ratio when the events where normally distributed. For geometrically distributed events with compression ratio 0.5 the packet delivery ratio deteriorates 1.5% for independent coalescing and 0.5% for simple coalescing. Although, for uniformly distributed events the data delivery ratio with 0.5 compression, improves by 2.2% for independent coalescing and 1.8% for simple coalescing.



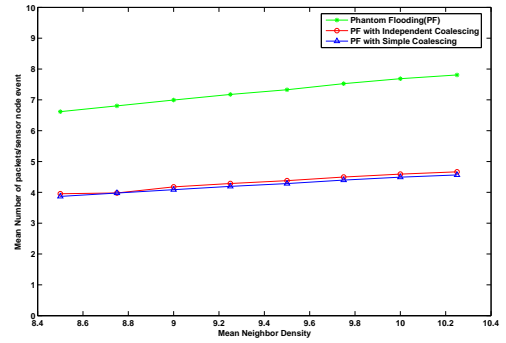
(a) Normally distributed events, compression ratio=0.25



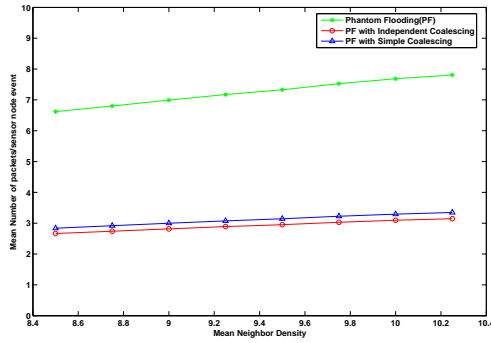
(b) Normally distributed events, compression ratio=0.5



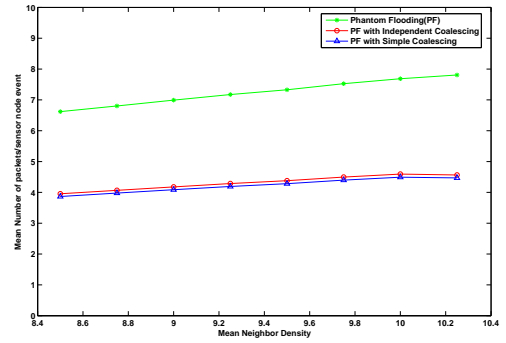
(c) Geometrically distributed events, compression ratio=0.25



(d) Geometrically distributed events, compression ratio=0.5



(e) Uniformly distributed events, compression ratio=0.25



(f) Uniformly distributed events, compression ratio=0.5

Figure 2: Mean Number of Packets sent/recvied as a function of Mean Neighbor density for normally, geometrically and uniformly distributed events, compression ratio = 0.25/0.5

From the results above we can conclude that there is minute deterioration of the packet delivery ratio of the underlying routing algorithm when embedded with coalescing algorithms for normally and geometrically distributed events. But we also observe that the packet delivery ratio of underlying algorithm is enhanced for uniformly distributed events with 0.5 compression ratio.

6.3.1 Energy Requirements

We use mean number of packets per node per event to measure the energy requirements of the phantom flooding and phantom flooding embedded with coalescing algorithms.

The figure 2 below shows the mean number of packets per node per event in the sensornet with phantom flooding, phantom flooding embedded with coalescing algorithms as a function of Mean neighbor density for normal, geometric and uniformly distributed events, for 0.25 and 0.5 compression ratios respectively.

From the figures above we observe that when independent coalescing is used for contextual privacy with a compression ratio of 0.25, It decreased the energy requirement per node per event by nearly 2.5 times. When independent coalescing is used for contextual privacy with a compression ratio of 0.5, It decreased the energy requirement per node per event by nearly 1.67 times. Similarly, When simple coalescing is used for contextual privacy the energy requirements per node per event is decreased by nearly 2.27 and nearly 1.71 times respectively.

Overall, in this scenario we showed the results to prove the correctness of phantom flooding we implemented and also we showed that the reliability of the underlying routing algorithm is deteriorated slightly by embedding coalescing algorithms. But we showed that coalescing algorithms offer huge energy savings and also we show that these coalescing algorithms provide much better contextual privacy then underlying spatial obfuscation technique in the following scenarios. Hence we argue that we can afford to lose minute reliability of underlying routing algorithm for better energy savings and better contextual privacy. We also observe from the above results that the number of packets generated per node per event in a sensornet is independent on the distributions of inter-arrival-times of events. Hence in the following experiments we restrict our experiments to normally distributed events.

6.4 SIMULATION RESULTS: SIMULATION SCENARIO 2

In this scenario, we evaluate the performance of coalescing algorithms against a single adversary. The performance metrics we use to define privacy are safety period [10] and capture probability [10].

6.4.1 Adversary Model 1

In this scenario we compare the safety period of phantom flooding and phantom flooding embedded with independent and simple coalescing. The figure 3 shows the safety period as a function of source-sink distance. The figure shows the privacy improvement due to coalescing using the compression ratio 0.25. From the figure we can see that the independent coalescing increases the safety period by at least 6 times. The simple coalescing increases the safety period by at least 5 times.

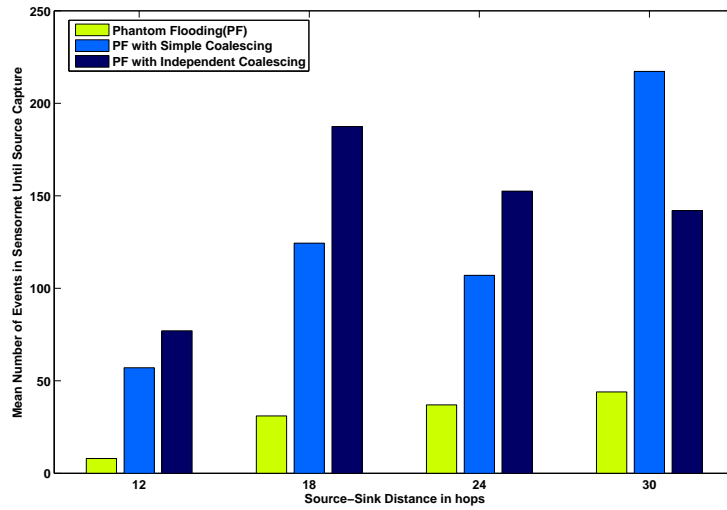


Figure 3: SafetyPeriod as a function of Source-Sink Distance in Hops, Number of Adversaries=1

The table 3 below summarizes the capture probability of phantom flooding and phantom flooding with simple and independent coalescing at various source-sink distances. We see that at larger source-sink distances simple coalescing decreases the capture probability drastically but independent coalescing does not effect the capture probability.

Table 3: Capture Probability of sink against single adversary, along various hop distances

	300	450	600	750
Phantom Flooding	100	100	100	100
PF with Independent Coalescing	95.8	100	100	100
PF with Simple Coalescing	96.2	72.7	40.7	53.3

6.5 SIMULATION RESULTS: SIMULATION SCENARIO 3

In this scenario we evaluate the performance of coalescing algorithms against a various sizes of adversarial network, i.e various number adversarial nodes in an adversarial network.

6.5.1 Adversarial Network Model 1 with two adversary nodes

In this scenario we place one adversary at the sink and the other adversary is symmetrically placed at the opposite corner of the sink. In this scenario we compare the safety period of phantom flooding and phantom flooding embedded with independent and simple coalescing. The figure 4 shows the safety period as a function of source-sink distance. The figure shows the privacy improvement due to coalescing using the compression ratio 0.25. The figure shows that the independent coalescing increases the safety period by at least 6 times. The simple coalescing increases the safety period by at least 5 times. We see that the as the power of adversary increases the performance of phantom flooding deteriorates quickly, unlike phantom flooding with independent and simple coalescing.

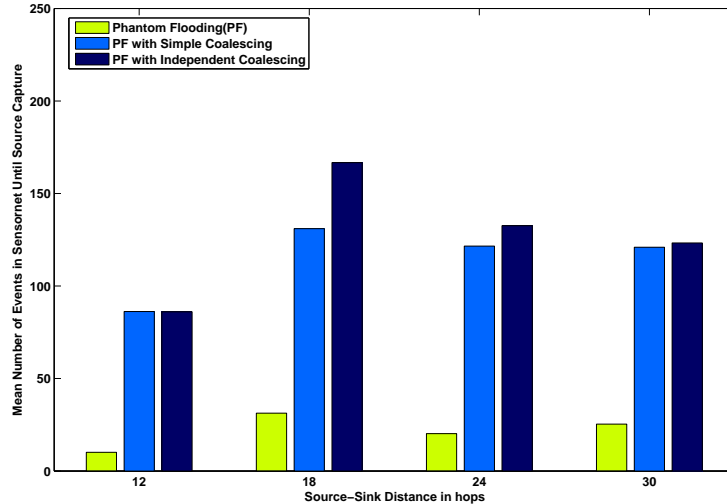


Figure 4: SafetyPeriod as a function of Source-Sink Distance in Hops, Number of Adversaries=2

The table 4 summarizes the capture probability of phantom flooding and phantom flooding with simple and independent coalescing at various source-sink distances. We see that at larger source-sink distances independent coalescing and simple coalescing decreases the capture probability reasonably well even with 2 adversaries.

Table 4: Capture Probability of sink against two adversaries, along various hop distances

	300	450	600	750
Phantom Flooding	100	100	100	91.7
PF with Independent Coalescing	100	91.2	80	84.6
PF with Simple Coalescing	97.1	84.6	76.5	91.7

6.5.2 Adversarial Network Model 1 with four adversary nodes

In this scenario we place one adversary at the sink and the other adversary is symmetrically placed at the opposite corner of the sink. In this scenario we compare the safety period of phantom flooding and phantom flooding embedded with independent and simple coalescing. The figure 5 below shows the safety period as a function of source-sink distance. The figure shows the privacy improvement due to coalescing using the compression ratio 0.25. The figure shows that the independent coalescing increases the safety period by at least 5 times. The simple coalescing increases the safety period by at least 4 times. We also see from the figures that the performance of phantom flooding without coalescing is really poor, averaging at a safety period of 10-12.

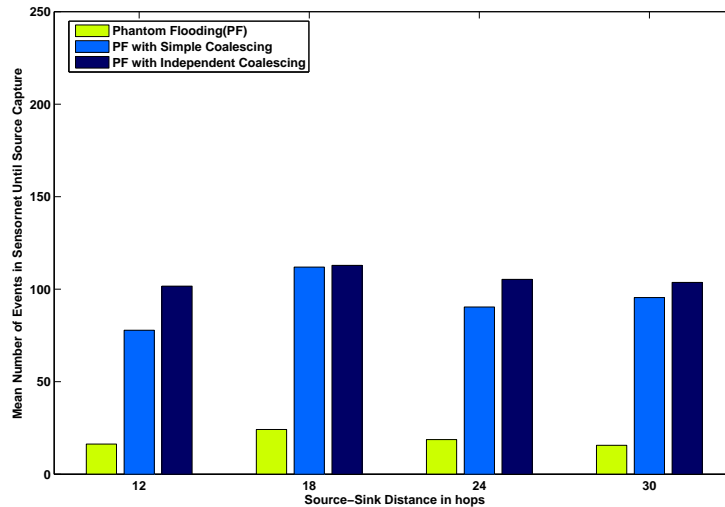


Figure 5: SafetyPeriod as a function of Source-Sink Distance in Hops, Number of Adversaries=4

The table 5 below summarizes the capture probability of phantom flooding and phantom flooding with simple and independent coalescing at various source-sink distances. We see that at larger source-sink distances independent coalescing decreases and simple coalescing decreases

the capture probability reasonably well. But we can see from the table that usually the power of adversary is proportional to the capture probability.

Table 5: Capture Probability of sink against four adversaries, along various hop distances

	300	450	600	750
Phantom Flooding	100	100	100	92.6
PF with Independent Coalescing	98	92	100	92
PF with Simple Coalescing	100	100	100	87

6.5.3 Adversarial Network Model 1 with eight adversary nodes

In this scenario we place one adversary at the sink and the other adversary is symmetrically placed at the opposite corner of the sink. In this scenario we compare the safety period of phantom flooding and phantom flooding embedded with independent and simple coalescing. The figure below shows the safety period as a function of source-sink distance. The figure 6 shows the privacy improvement due to coalescing using the compression ratio 0.25. The figure shows that the independent coalescing increases the safety period by at least 5 times. The simple coalescing increases the safety period by at least 4 times. We also see that, even with a very powerful adversary the coalescing techniques offer reasonable contextual privacy unlike phantom flooding without any coalescing, which deteriorates very fast with increase in power of adversary.

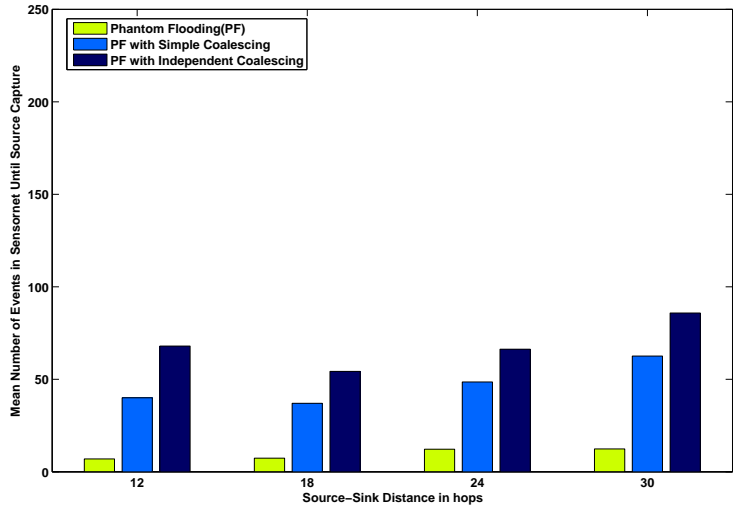


Figure 6: SafetyPeriod as a function of Source-Sink Distance in Hops, Number of Adversaries=8

The table 6 below summarizes the capture probability of phantom flooding and phantom flooding with simple and independent coalescing at various source-sink distances. We see a very powerful adversary would make the probability of capture 100% even when using independent/simple coalescing.

Table 6: Capture Probability of sink against eight adversaries, along various hop distances

	300	450	600	750
Phantom Flooding	95.5	100	95.5	100
PF with Independent Coalescing	100	100	100	100
PF with Simple Coalescing	100	100	97.4	100

7 CONCLUSION AND DIRECTION OF FUTURE RESEARCH

This thesis introduces in situ coalescing, a technique for obfuscating temporal contexts of sensing events in a wireless sensor network environment. By exercising this strategy in a time insensitive asset monitoring application, we can perform lazy reporting of sensed data as a measure for protecting contextual privacy. Our work is complementary and continuative to the existing techniques for obfuscating spatial contexts, such as phantom flooding [5, 10]. The proposed line of research to enhance contextual privacy in sensornets, has the advantage of using sensor data coalescing that results in power saving due to the infrequent active cycles in radio transmission and the diminution in total message size that needs to be transmitted. Through our simulation experiments, we have shown that the coalescing techniques, if used in conjunction with phantom flooding, partially compensates for the additional power expended in the directed random walk and flooding phases of phantom flooding while enhancing the safety period offered by phantom flooding greatly. We also demonstrated, how fast the efficiency of phantom flooding deteriorates with the increase of power of adversary. But at the same time we demonstrated that when phantom flooding is embedded with coalescing, the safety period still remains relatively high enough to protect the context. Moreover, we also showed that capture probability is considerably decreased with coalescing algorithms.

An impending task would be to extend our simulation experiments for the purpose of studying the effectiveness of coalescing approaches embedded with various spatial-obfuscation techniques. Which would help us build an integrated framework that can look for tradeoff in combining the right extent of temporal techniques and spatial techniques. In addition, this framework must be aware of the application semantics that specifies the need for responsiveness

and contextual privacy. We intend to study this combination in future and determine if it leads to a more effective protocol for supporting contextual privacy in an adversarial environment.

8 ACKNOWLEDGEMENTS

This research was supported in part by NSF under grant:CNS-0551734 and the information systems and security systems (ISSS) research cluster at NMSU.

REFERENCES

- [1] D. Chaum. Untraceable electronic mail, return addresses, and digital pseudonyms. In *Communications of the ACM*, vol. 24, no. 2, pp. 84-88, February 1981.
- [2] T. M. Cover and J. A. Thomas. *Elements of Information Theory*. John Wiley, 1991.
- [3] J. Deng, R. Han, and S. Mishra. Countermeasures Against Traffic Analysis Attacks in Wireless Sensor Networks. In *1st IEEE/CreateNet Conference on Security and Privacy for Emerging Areas in Communication Networks (SecureComm 2005)*, Athens, Greece, pp. 113-124, September 2005.
- [4] L. Ekroot and T. M. Cover. The entropy of a randomly stopped sequence. In *IEEE Transactions on Information Theory*, vol. 37, no. 6, pp. 1641-1644, November 1991.
- [5] P. Kamat, Y. Zhang, W. Trappe and C. Ozturk. Enhancing source-location privacy in sensor network routing. In *Proc. 25th Int'l Conference on Distributed Computing Systems (ICDCS 2005)*, Columbus, Ohio, pp. 599-608, June 2005.
- [6] J. N. Kapur and H. K. Kesavan. *Entropy Optimization Principles with Applications*. John Wiley, 1992.
- [7] K. Martinez, J. K. Hart and R. Ong. Environmental sensor networks. In *IEEE Computer*, vol. 37, no. 8, pp. 50-56, August 2004.
- [8] The ns-2 network simulator. <http://www.isi.edu/nsnam/ns>.
- [9] C. Ozturk, Y. Zhang, W. Trappe and M. Ott. Source-location privacy for networks of energy-constrained sensors. In *Proc. 2nd IEEE Workshop on Software Technologies for Future Embedded and Ubiquitous Systems, (WSTFEUS 2004)*, Vienna, Austria, pp. 68-72, May 2004.
- [10] C. Ozturk, Y. Zhang and W. Trappe. Source-location privacy in energy-constrained sensor network routing. In *Proc. 2nd ACM Workshop on Security of Ad Hoc and Sensor Networks (SASN 2004)*, Washington, D.C., pp. 88-93, October 2004.
- [11] T. Park and K. Shin. LiSP: A lightweight security protocol for wireless sensor networks. In *ACM Transactions on Embedded Computing Systems (TECS)*, vol. 3, no. 3, pp. 634-660, August 2004.
- [12] A. Perrig, J. Stankovic and D. Wagner. Security in wireless sensor networks. In *Communications of the ACM*, vol. 47, no. 6, pp. 53-57, June 2004.
- [13] A. Perrig, R. Szewczyk, D. Tygar, V. Wen and D. Culler. SPINS: security protocols for sensor networks. In *Wireless Networks*, vol. 8, no. 5, pp. 521-534, September 2002.
- [14] M. G. Reed, P. F. Syverson and D. M. Goldschlag. Anonymous connections and onion routing. In *IEEE Journal on Selected Areas in Communications*, vol. 16, no. 4, pp. 482-494, May 1998.
- [15] S. M. Ross. *Stochastic Processes*. 2nd edition, John Wiley, 1996.
- [16] R. Szewczyk et al. Habitat monitoring with sensor networks. In *Communications of the ACM*, vol. 47, no. 6, pp. 34-40, June 2004.
- [17] Y. Ouyang, Z. Le, G. Chen, J. Ford and F. Makedon. Entrapping Adversaries for Source Protection in Sensor Networks. In *Proc. 2006 International Symposium on on World of Wireless, Mobile and Multimedia Networks (WOWMOM '06)*, Washington, D.C., pp. 23-34, June 2006.