Set Difference Attacks in Wireless Sensor Networks

Tulio de Souza[1], Joss Wright[2], Piers O'Hanlon[2], and Ian Brown[2]

[1] Department of Computer Science, University of Oxford, UK
tulio.de.souza@cs.ox.ac.uk
[2] Oxford Internet Institute, University of Oxford, UK
{joss.wright,piers.ohanlon,ian.brown}@oii.ox.ac.uk

Abstract. We show that existing proposed mechanisms for preserving the privacy of reported data values in wireless sensor networks are vulnerable against a simple and practical form of attack: the *set difference attack*. These attacks are particularly effective where a number of separate applications are running in a given network, but are not limited to this case. We demonstrate the feasibility of these attacks and assert that they cannot, in general, be avoided whilst maintaining absolute accuracy of sensed data. As an implication of this, we suggest a mechanism based on perturbation of sensor results whereby these attacks can be partially mitigated.

Keywords: privacy, privacy-preserving, wireless sensor network, data perturbation, differential privacy, privacy-preserving data aggregation.

1 Introduction

Wireless sensor networks are increasingly being deployed to monitor a variety of real-world environments and processes. Initially designed for military applications such as battlefield monitoring or perimeter security, wireless sensor networks are now being used to monitor industrial processes, environmental pollution, marine- and land-based ecosystems, and stock control, as well as many other purposes.

The data gathered by wireless sensor networks can in many cases be sensitive, either when considered in isolation or when combined with other data. Where individuals and their actions are monitored by a wireless sensor network we desire, or may even be legally required [5], to ensure adequate protection measures for personally sensitive data. Even when data is not directly sensitive, it is good privacy and security hygiene to prevent unnecessary dissemination of readings from individual sensor nodes.

In practice, wireless sensor networks occur with varying degrees of complexity [15]. These networks can be roughly classified according to their structure, either as standalone, multi-application or federated multi-application networks.

The simplest wireless sensor networks have tended to be standalone systems running a bespoke application that defined both the constituent nodes and all other aspects of the network. In such a deployment, hardware requirements are tailored to fit the needs of the application in question, with the application exploiting all aspects of the network. This structure remains common today.

Increasingly, however, wireless sensor networks are being deployed in a multi-application structure comprising nodes running a common middleware that allows one or more applications to run on the same infrastructure. The use of middleware offers a flexible and standardized abstraction of the low-level characteristics of the hardware, allowing data collected by each node to serve a number of applications. This increases the range of uses for a given deployment, but also has the potential to raise privacy or security concerns.

The sharing model can be extended further by allowing *federation* of the infrastructure. A federated multi-application network allows different entities to run applications across the same set of nodes, sharing resources between multiple stakeholders. This provides an economic benefit, and can lead to longer-term deployments offering a range of sensing options, but also raises even greater privacy concerns for those individuals in the sensing environment [11].

To date, research in wireless sensor network privacy has focused largely on privacy-preserving data aggregation (PPDA) protocols that protect the data collected in sensor nodes against outside observers, or limited malicious network participants. Importantly, existing protocols have focused almost solely on standalone networks, without consideration for the more complex multi-application and federated networks.

In this work we are chiefly concerned with protecting, or conversely learning, individual readings from nodes in a wireless sensor network. Specifically, we are concerned with the potential to derive individual sensor node readings in a range of network structures, but we focus on networks that support multiple applications, even in the presence of existing privacy-preserving protocols.

The remainder of the paper is structured as follows. Firstly, we define our model and underlying assumptions, and introduce the notion of the *set difference attack*. We then explore the capabilities and goals of existing privacy-preserving data aggregation protocols, describe in detail how they fail to protect against these attacks, and analyse the potential for these attacks to function in practical deployments. Finally, we propose an initial approach towards mitigating these attacks, and explore its implications for data collection in sensor networks.

2 System and Attacker Model

We are concerned with wireless sensor networks in which multiple stakeholders deploy applications that aggregate information provided by nodes in the network.

More formally, we consider a wireless sensor network $\mathcal{W}$ as being comprised of a set of discrete sensor nodes $\mathcal{S} = \{s_1, s_2, ..., s_n\}$ along with a function mapping nodes to their reported readings modelled as simple natural numbers: $\mathcal{V} : \mathcal{S} \rightarrow \mathcal{N}$. Users query some subset of sensor nodes, corresponding to those running some application, and receive a simple addition of the individual sensor values:

$\mathcal{V}(\mathcal{A}) \mid \mathcal{A} \subseteq \mathcal{S}$; we further assume that both the set of nodes comprising a given application and the aggregate results of any queries are known[1].

Our goal is to protect or, adversarially, to learn the reading of any individual sensor: $\mathcal{V}(\{s\}) \mid s \in \mathcal{S}$.

We assume that applications aggregate a known subset of $\mathcal{S}$, reporting only an aggregate value. Intrinsically, we assume some lower limit on the size of the set $\mathcal{A} \subseteq \mathcal{S}$ in order to prevent trivially requesting the value of an individual node. We will show later how this simple defence is ineffective.

We consider two attacker models based on the standard *global passive attacker* commonly used in the field of privacy-enhancing technology research. This attacker is able to observe, but not decrypt, traffic passing between nodes but cannot alter, delay or drop communications; nor can this attacker compromise an individual node directly.[2]

We will focus on this first, truly passive, attacker restricted simply to observing the aggregate readings of applications, however the nature of our system model also naturally lends itself towards a *partially active* attacker that may deploy one or more applications subject to the limitations inherent in the system. We distinguish this from a truly active attacker in that this attacker may not drop or delay communications. These attackers correspond, respectively, to a non-stakeholder that queries the aggregate results of applications deployed by others in the network, and to a stakeholder with the ability to deploy their own applications on demand but who will not engage in openly malicious behaviour.

Further, for the current work we focus on a static moment and will not analyse in detail the potential effects of long-term analysis of sensed values. We will, however, make some mention of the effects of timing with respect to node availability in subsequent sections, but leave detailed investigation of this for future work.

The model we have described here represents recent research in federated wireless sensor network design, for example the work of Leontiadis et al. [12].

3 Set Difference Attacks

A set difference attack exploits the intersections between the sets of sensors comprising applications to discover scenarios in which individual nodes, or small clusters of nodes, are isolated.

The simplest form of this attack is demonstrated in Figure 1. The node coverage of two small applications is delineated by the light-grey regions. The first application covers the set $\{b, d, e\}$, and the second the set $\{b, c, d, e\}$. An application querying aggregate results from these two applications can trivially subtract

[1] While this may seem to place a great deal of information in the hands of potential attackers, it is a reasonable representation of existing wireless sensor network platforms. It should also be noted that the attacks we will describe remain feasible with greatly reduced, or more localized, information.

[2] Note that this attacker differs from the common Dolev-Yao attacker in security protocol literature in that it cannot affect messages in transit.

the aggregate of the first application from the aggregate of the second in order to learn the exact value of node c.

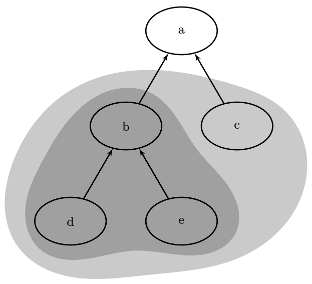

Fig. 1. Simple set differences in a WSN

This form of attack has some similarities to known attacks in statistical databases, known as *tracker attacks* [1], as well as to attacks against mix-based anonymous communications systems in the form of $(n-1)$ attacks [8]. In Section 5 we will explore more complex scenarios in which these attacks apply.

An interesting feature of these attacks is that they rely only on consideration of aggregate values reported to a sink, and thus make no attempt to read data as it passes across the network. Crucially, as we will demonstrate, this makes these attacks applicable against most well-known families of privacy-preserving protocols for wireless sensor networks proposed in the literature.

Having introduced the set difference attack, we will now describe the most common approaches towards protecting privacy of individual sensor node readings in wireless sensor networks before showing how the attack applies against these protocols.

4 Privacy-Preserving Protocols in Wireless Sensor Networks

4.1 Goals

Privacy-preserving protocols in wireless sensor networks aim to preserve the privacy of individual nodes against some combination of the *sink*, the node that aggregates values reported by other nodes, and against other nodes in the network. Different approaches have tended to focus on some combination of these, with mixed results.

The protocols shown here make various trade-offs between communication complexity, computational requirements, integrity of data, and security.

4.2 Clustering

Privacy-preserving clustering, illustrated in Figure 2, functions by forming disjoint subsets of nodes, each of which calculates an aggregate sum of their data before it is sent to the sink. A variety of approaches are possible to achieve this aggregation, but a popular approach [9] makes use of a variation on the Average Salary Problem. This algorithm, a simple instance of the more general secure function evaluation problem, allows nodes to sum their individual values without leaking any more information than the aggregate itself. The desired effect is that neither the sink, nor any node in the cluster, can learn the exact value of any individual node unless $(n-1)$ nodes in a cluster of size n collude.

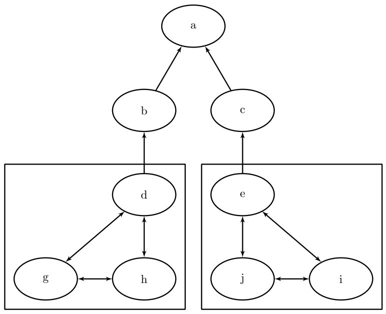

Fig. 2. Private clustering in WSNs

An advantage of the clustering approach is that it prevents both the sink and any individual nodes in the network from learning any single node's values, at the expense of the bandwidth required to form clusters and perform the secure data aggregation.

4.3 Slicing

Slicing, introduced in [9] and then expanded in [10], chiefly aims to prevent individual nodes in the network from learning the values reported by any other nodes.

To achieve this, a node divides its values into a number of randomly-sized slices and selects multiple paths through the network, as illustrated in Figure 3. Each slice is sent via a different path, and added to the total sum calculated by each intermediate node, which acts as an aggregator until the value reaches the sink. The number of paths that each node sends its data acts as a configurable parameter to the required privacy level. This simple mechanism aims at providing confidentiality against other nodes in the network as well as the sink.

4.4 Privacy Homomorphisms

Privacy homomorphism uses the well-known homomorphic properties of certain public-key encryption systems to aggregate data in transit without revealing individual values. Again, this mechanism provides protection against external attackers and malicious nodes in the network, but does not prevent the sink from learning individual values.

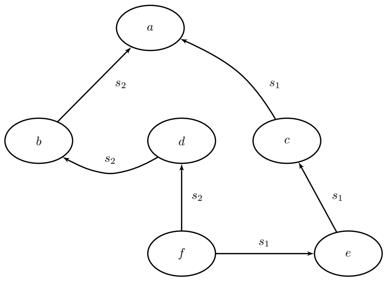

Fig. 3. Private slicing showing the path of two private data 'slices' travelling across the network from f to a

Homomorphic encryption schemes allow manipulation of message plaintexts via the corresponding encrypted ciphertexts, enabling operations such as aggregation, or summation, of messages to be performed without decryption. Many well-known encryption schemes allow restricted homomorphic operations; in the Paillier scheme, for example, the multiplication of two ciphertexts under the same public key will decrypt to the summation of corresponding plaintexts, whilst raising one ciphertext to the power of another will decrypt to the product of the plaintexts.

Gentry [7] presented a fully homomorphic encryption scheme, allowing for arbitrary operations to be performed on ciphertexts. Whilst the original scheme was extremely computationally expensive, several improved schemes have already been suggested. In practice, however, even restricted homomorphism provides powerful and practical tool for privacy-preserving protocols.

In a wireless sensor network, therefore, nodes simply encrypt their values to the public key of the sink. As the message is relayed through the network, nodes can aggregate the value of any received messages simply by aggregating the ciphertexts, as demonstrated in Figure 4. Crucially, this protects the values of any individual message from being learnt by any party except the sink.

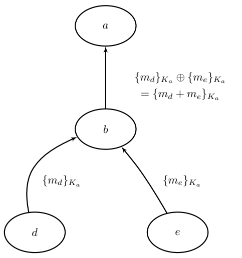

Fig. 4. Homomorphic encryption in WSNs. Values are aggregated in encrypted form at each node.

5 Set Difference Attacks in Detail

The set difference attack seeks to isolate nodes from aggregates in order to breach the privacy of their data. In practice, this can be achieved in one of two ways.

Firstly, the segmentation of the network caused by multiple applications running across disparate set of nodes can be exploited. An attacker therefore combines aggregate values of multiple applications in order to isolate single nodes. It is this approach on which we focus in the current work.

Secondly, an attacker can exploit the participation of nodes in aggregates taken at different moments in time. If nodes cannot be guaranteed always to report their values, then the aggregate value of an aggregate may include or exclude certain nodes when queried at different times. This behaviour is extremely likely to result in a set difference attack, as the set of nodes being queried is likely to remain largely the same.

These two approaches can be employed in isolation, or combined by an attacker. If the attacker can learn predictable patterns of node uptimes across the network, or can observe that certain groups of nodes are more likely to be clustered in applications, the effectiveness of the attack is increased.

While the example set difference attack shown in Figure 1 is relatively simple, the attack itself is surprisingly powerful and hard to avoid. In addition to the simple isolation of a node via finding an appropriately-sized subset, four additional cases are worthy of mention.

5.1 Isolated Cluster

Trivially, the set difference attack allows us to reveal the aggregate value of an isolated *cluster* rather than an individual node. While this is not a privacy risk equivalent to the leakage of an individual node value, the leaking of the aggregates of a small set of nodes may still be in violation of the privacy goals of the system.

5.2 Combined Subsets

Although the most basic form of set difference attack comes from observing a subset of size $n-1$ of a given set of size n, it is of course possible for the subset to itself be the union of a number of disjoint subsets as illustrated in Figure 5.

This possibility greatly increases the likelihood of observing a successful set difference attack. Observed aggregates can be stored by an attacker and combined whenever new appropriate aggregates are found. Of course, this application of the attack is highly time-dependent.

5.3 Total Set Coverage

In general, set difference attacks are not possible where observed subsets overlap, as this includes multiple unknown values in the combined aggregates. It is possible, however, to calculate values through gathering complete collections of

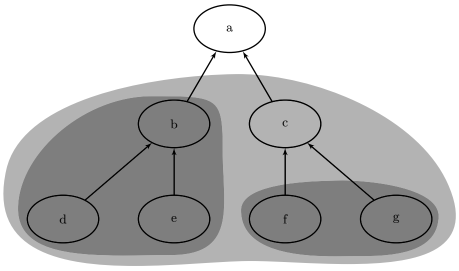

Fig. 5. A set difference attack combining multiple disjoint subsets

sets that intersect on all but one of their elements. By gathering every possible subset of size $n - 1$ from a set of size n, we can derive *all* individual values that comprise the set. The aggregate values reported for each subset form a simple system of simultaneous equations that can be solved for each individual value.

The difficulty of performing this attack relies on the size of the subsets that we observe, as we require all $\binom{n}{n-1}$ subsets of the observed subset of size n. While we will not perform a detailed analysis of the likelihood of this scenario, it relates to the well-known *coupon collector's problem* [4] in which a collector seeks to obtain a complete collection of a set of coupons, one of which is randomly included with each purchase of a given product. It is known that the number of purchases required before obtaining the entire set of coupons is of the order $n\ log(n)$, where n is the number of coupons in the set. For large networks, this scenario quickly becomes highly unlikely, however it may be practical in smaller networks or those networks where applications are likely to sample from small sets of related nodes.

5.4 Attack Recursion

The result of a successful set difference attack provides information to an attacker that can lead to further successful attacks. By learning the value of an individual node, or of a small subset of nodes, an attacker can remove that node's value from any observed aggregates in the network. This may itself reveal further isolated subsets that can themselves compromise further sets. As such, the attacker can potentially 'recurse' through several further attacks once any one attack has succeeded.

6 Attacking Existing Protocols

Existing approaches to protecting privacy in wireless sensor networks focus, to varying degrees, on manipulating data as it flows from a sensor to a sink. Clustering approaches aggregate data by combining values that are then forwarded in aggregate form. Slicing approaches split data unpredictably and randomly re-route individual portions along different paths. Privacy homomorphism encrypts data in such a way that it can be unobservably aggregated in transit. The set difference attack, however, is entirely agnostic with respect to the flow of data; instead it operates purely through examination of the final aggregate, undermining the assumptions of existing protocols and therefore rendering them vulnerable.

Clustering, in particular, may actively aid in the application of a set difference attack. As presented in [9], the choice of node clusters is random. A result of this is that multiple requests by an application are likely to result in the selection of different clusters. These, in turn, can directly cause the isolation of nodes in precisely the way envisioned in our original statement of the attack.

Slicing approaches and solutions based on homomorphic encryption share similar patterns of failure. The values of each node are protected, or at least obscured, whilst in transit, however the results are still accurately aggregated by the application. Whilst the existing protocols do provide some measure of protection against the specific threat model of an adversary that seeks to learn values in transit, they are ineffective against the attacker described in Section 2.

Ultimately, it is the requirement for accurate data reporting that results in the success of the set difference attack, and it is therefore this feature of the network that must be addressed by protocols in order to prevent the attack.

6.1 Node Availability

As we have mentioned, it is possible to perform a set difference attack through node availability rather than overlapping applications. In this case, an application that has a known, fixed set of nodes, but for which certain nodes are not always available, the absence or presence of individual nodes can clearly lead to similar attacks. Most notably, this attack will be effective even in single-application networks.

The inclusion of a time dimension in the attack clearly adds a layer of sophistication to the attack. If the availability of certain nodes is predictable, queries can be specifically targeted to take advantage of this data. Interestingly, an individual node has little power to prevent this attack in the general case, as it will be offline when the attack effectively occurs.

A slightly more nuanced version of this attack, which we leave for future work, comes from the predictability of individual nodes over time. Clearly, certain types of sensor readings will vary predictably with time, such as light levels during the day. This can lead to predictable patterns of data being reported for each node. A more sophisticated variant of the attack would be to infer variations between nodes due to the predictable variations in aggregate reports. Similar concepts have been suggested in the context of tracking of users in online anonymization services [13], however we will not consider this potential further in the current work.

7 Feasibility of Set Difference Attacks

To investigate the feasibility of the set difference attack in practice, we adopt a simulation-based approach, employing abstract networks of varying size based on the system model of Section 2.

For each experiment, randomly-sized subsets of the network were repeatedly drawn at random. Each subset was stored and compared against all previously-drawn sets, individually and in additive and subtractive operations, to determine if a set difference attack had become possible. An attack was considered to have occurred as soon as any individual node could be isolated due to the combination of any number of previously-drawn sets. Sets were drawn continually until the attack succeeded, whereupon the number of sets drawn was recorded. To prevent trivial attacks, subsets were restricted to being of cardinality three or greater, up to the size of the network. To ensure a sufficiently low error margin for the mean, experiments were repeated in the order of one thousand times for each network size.

As a practical example of a successful simulated attack, consider a network of five nodes, $\mathcal{S} = \{a, b, c, d, e\}$, in which each node is equally likely to be selected. During a particular simulation run, three subsets were drawn: $\mathcal{A}_1 = \{a, c, e\}$, $\mathcal{A}_2 = \{a, b, c, d, e\}$ and $\mathcal{A}_3 = \{a, b, d\}$. The isolation of a node occurs by subtracting the aggregate of $\mathcal{A}_2$ from that of $\mathcal{A}_1$, which is then summed with the aggregate result of $\mathcal{A}_3$. This sequence of operations will result in isolating the reported reading of node a. Note that both operations, additive and subtractive, take place over the aggregate result of a query sent towards a subset of nodes, and not as subset operations.

The results of the simulation, showing the mean number of sets drawn before a successful attack, are presented in Figure 6.

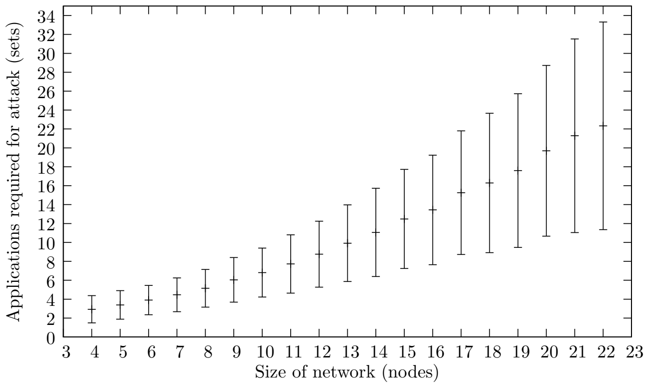

Fig. 6. Mean average and sample standard deviation of randomly-chosen sets required in networks of varying size before a successful set difference attack

As can be seen, the mean number of subsets required before isolating a single node is relatively low in the simulated networks, typically being lower than the number of nodes. The growth of the function does, however, appear to be more than linear, as might be expected due to the rate of increase of possible subsets. While this suggests that extremely large networks may not be easy targets for the set difference attack, networks of the size commonly seen in practice may well be vulnerable. Despite this it is worth noting that the lower bound for the required number of sets remains two, and simulation demonstrated such attacks occurring in practice for each network size that was tested. Due to space considerations, we leave a more detailed analysis of these results for future work.

Calculating the appropriate sets required to conduct an attack is itself extremely computationally expensive. As each new set is drawn, it must be combined with all existing sets, both in an additive and subtractive sense, to determine if an attack has been successful. The stored sets, and the number of comparisons required, grow exponentially. There are various optimizations to reduce the number of sets that must be stored and compared, and various ways to exclude sets that cannot take part in a successful attack, however the underlying complexity of the problem cannot be avoided.

For the sake of practicality, it will be possible to take a heuristic approach towards discovering set overlaps that, despite missing a proportion of successful attacks, will still result in isolating individual nodes. It is also the case that, as we have discussed, real-world networks present time constraints on the freshness and availability of sensor readings. This will present challenges to the attacker in discovering appropriate sets during a given time window, but will also greatly reduce the complexity required to perform the attack.

8 Preventing Set Difference Attacks

As has been demonstrated, existing protocols cannot protect node-level privacy against the set difference attack under reasonable assumptions. This is largely due to their reliance purely on data aggregation to provide privacy guarantees at the node level. In this section, we will consider the use of *data perturbation* to provide effective privacy guarantees, and examine the accuracy tradeoff that these approaches cause.

8.1 A Note on Fixed Clustering

Before we discuss data perturbation it is worth first mentioning one potential avenue of protection against set difference attacks, and explaining why this approach is unlikely to be of great use.

One approach that initially seems attractive for protecting against this form of attack is to enforce fixed-size clusters, or fixed size applications, and ensure the subsets of nodes resulting from these are either entirely disjoint or entirely equal. By doing so, individual nodes cannot be isolated, and thus the attack fails.

There are two major problems with this approach. Firstly, it places unreasonable constraints on applications in a multi-application or federated network. Specific deployments are likely to require specific node coverage, and the inability to choose other than a given fixed topology for applications could seriously hinder the flexibility of the network.

More seriously, this approach still cannot protect against attacks due to unavailable nodes. As is mentioned in Section 6.1, set difference attacks can arise from both predictable patterns of node availability, and potentially from predictable patterns of sensor readings. Neither of these factors will be affected by fixed-size clustering, and thus cannot provide full protection against the attack. We will therefore focus on other, fundamentally different, approaches.

8.2 Data Perturbation

To protect against a set difference attack, we propose applying random noise to sensor readings. The purpose of this is to prevent the individual value reported by a node from being meaningful even if it can be isolated by the attack. Clearly, for some applications, this *data perturbation* approach can cause an unacceptable level of inaccuracy in aggregate results. In such cases, the risks of attack must be weighed against the requirement for accurate data.

Sensor nodes can effectively obscure their data by adding random noise drawn from an appropriately-scaled symmetric probability distribution with mean 0 to their reported readings. To protect readings effectively, the standard deviation of the distribution in question should be chosen according to the possible range of values for the given reading type. Due to Chebyshev's inequality, this ensures that the value reported by a node, including noise, effectively covers a range of values that could be reported by the node with high probability. In the next section, we will discuss a well-known method for selecting privacy-preserving noise optimally according to the *differential privacy* guarantee of Dwork [2], where we will also discuss the notion of data perturbation in more detail.

Usefully, combining multiple readings and their associated random noise causes the aggregate noise to converge rapidly towards zero as the number of nodes increases, due to the weak law of large numbers. The aggregate therefore tends towards greater accuracy as the number of nodes in a given application increases, making the data perturbation approach increasingly applicable as the network scales.

For noise drawn from a Gaussian distribution the sample mean, representing the aggregate noise reported from each sensor, is a good estimator of the true mean. The mean standard error of the sample mean, therefore, describes the expected inaccuracy incurred by this method of privacy-preserving data perturbation. To summarize:

For the sample mean:

$$\bar{X} = \frac{1}{n}\sum_{i=1}^{n} X_i \tag{1}$$

The mean standard error can be described as:

$$MSE(\bar{X}) = E((\bar{X} - \mu)^2) = \frac{\sigma^2}{n} \tag{2}$$

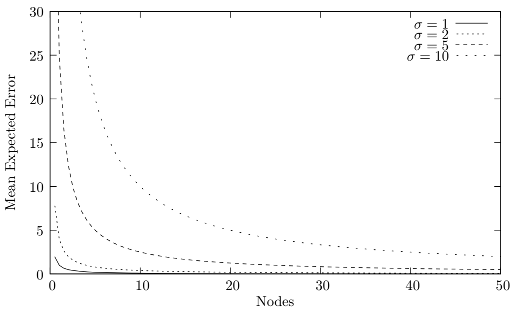

Fig. 7. Mean standard error (MSE) for various values of σ as application size increases

As can be seen from Figure 7, the expected error rapidly becomes small as the number of nodes in an application increases, even for relatively large values of σ.

Perturbation Alongside Other Mechanisms. It is important to note that the perturbation of data in the sense we have described above is largely orthogonal to the mechanisms surveyed in Section 4. As such it is entirely possible, and may indeed be advisable, for nodes to cluster, slice or encrypt their data in addition to perturbing their data. In particular, this approach has the potential to improve the node-level privacy even in situations where set difference attacks are not possible, and may add privacy properties that protect against other classes of attacker. We leave a fuller analysis of the combination of perturbation with other mechanisms for future work.

Having examined an informal approach to data perturbation, we will now discuss the more formal and optimal guarantees that can be provided by *differential privacy*.

9 Differential Privacy

The technique of gaining privacy in statistical aggregates through data perturbation is not new, and indeed represents a well-known approach that is the subject of much recent study. An important result in this area comes from Dwork [2], in which the concept of *differential privacy* is proposed. This technique aims to provide robust privacy guarantees through data perturbation, with a provably minimal addition of noise to the result of statistical queries.

The core of the differential privacy guarantee is that the existence or absence of a single record in a database should not cause a noticeable difference in the result of queries against that database. This is achieved by ensuring that databases that differ only in a single record are, in some sense, indistinguishable to any party able to make statistical queries against that database.

The purpose of this indistinguishability is to prevent an individual record from leaking useful information even in the presence of arbitrary, unknown *auxiliary information*. By ensuring that no single record is distinguishable in a statistical query, differential privacy ensures that any privacy breach involving statistical queries from a database could have occurred *without* the result of that statistical query.

More formally, differential privacy states that for any two databases D_1 and D_2 that differ only in a single data record, the result of a randomized statistical query should be almost equally probable for D_1 or D_2. Dwork's original statement of the guarantee provides that a randomised function $\mathcal{K}$ achieves ϵ-differential privacy if, for any two databases $\mathcal{D}_1$, $\mathcal{D}_2$ differing on at most one element, and all $S \subseteq Range(\mathcal{K})$:

$$\Pr[\mathcal{K}(\mathcal{D}_1) \in \mathcal{S}] \leq \exp(\epsilon) \times \Pr[\mathcal{K}(\mathcal{D}_2) \in \mathcal{S}]$$

where ϵ is a security parameter that allows security to be balanced against accuracy of results.

The probability of a given result is therefore within a small multiplicative factor regardless of whether D_1 or D_2 was queried. This ensures that the result of a statistical query cannot be used to determine with any certainty which database was queried. It thus becomes impossible for the existence or absence or a record, or its value, to be determined.

The differential privacy guarantee is extremely strong. As is clear from the definition, however, repeated queries against the same databases will reveal more information concerning the underlying probability distributions, and eventually allow the databases to be distinguished with high probability. Differential privacy therefore provides the concept of a 'privacy budget' that is partially exhausted with each query. Once that budget is exhausted, no further queries can be made against the database in question without violating the guarantee.

In practice, differential privacy is most commonly achieved by applying noise drawn from a Laplace distribution with mean 0 and standard deviation proportional to the *sensitivity* of the query and the strength of the guarantee, expressed as the probability of differentiating the two databases as a result of the query. This sensitivity, written Δf for some query function f, is the greatest value by which the result of the query can change according to the change of a single record:

$$\Delta f = max(f(D_1) - f(D_2))$$

for all D_1, D_2 that differ in at most one record. As an example, a simple count function on a database, which returns the number of records that meet a given constraint, has a sensitivity of 1, as the addition or deletion of a single record can alter the result of the count by at most 1. Clearly, for functions that have high sensitivity, such as the average height in centimetres of a small group of individuals, achieving the differential privacy guarantee may require unacceptably high costs in terms of accuracy.

Figure 8 illustrates the desirable property of the Laplace distribution for applying noise. The probability of the observed events a and b from the perspective of each probability distribution are within a small, fixed multiplicative factor, allowing each result to be convincingly drawn from either distribution.

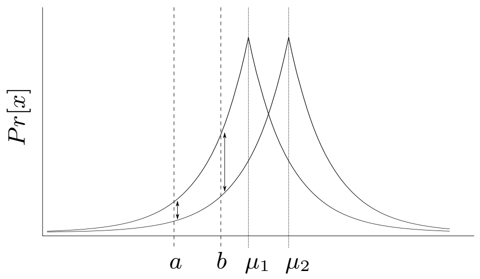

Fig. 8. Overlapping Laplace distributions, means μ_1 and μ_2, showing comparative probabilities of two values, a and b, drawn according to either distribution

A significant advantage of this differential privacy mechanism is that it is largely independent of the data itself, but provides its guarantees due to the nature of the query function.

9.1 Differential Privacy against Set Difference Attacks

Application of a differentially-private mechanism for protecting individual sensor node readings in WSNs functions similarly to the addition of random Gaussian noise as described above. The use of differential privacy, however, provides a number of attractive advantages over more *ad-hoc* methods. The guarantee provided by the mechanism gives provable privacy preservation for individual sensor nodes [2], as well as a number of attractive properties such as composability between multiple queries, at the cost of higher levels of inaccuracy. The generality of the method makes it applicable without reference to the data reported by the sensor node, relying instead on the query made by the application. Queries can be of arbitrary complexity, and are not restricted to simple functions such as counts or averages, although the noise associated with high sensitivity queries cannot be avoided. Despite this, the Laplace distribution has been shown in [3] to give a provable optimal level of noise, reducing inaccuracies in query results to the minimum required for a strong guarantee of privacy.

It is crucial to note that this framing of private data reporting is substantially different to the mechanism for differential privacy described by Dwork [2]. The original framing of differential privacy considers an accurate data store, corresponding to a sink in a wireless sensor network, that is trusted to hold and process an entire dataset. In our model, by contrast, we explicitly consider the sink as an adversary that we wish to prevent from learning individual data values, analogous to records in the database. As such, our model can best be conceived by considering each sensor node as analogous to the trusted database in Dwork [2]. Each node therefore represents a single-entry database that must, correspondingly, add a sufficient amount of noise to hide that entry. We rely on the aggregation of these single-entry databases to reduce the overall noise. The result of this is a higher level of noise than would be seen if the sink held accurate values, but with the advantage of preserving node-level privacy from all actors in the network.

10 Conclusions and Future Work

We have described the set difference attack, and shown that existing approaches to providing node-level privacy in wireless sensor networks are vulnerable to this attack under reasonable assumptions. Further, we have demonstrated that the attack is likely to be feasible in real-world networks. We propose that the weakness of existing privacy-preserving data protocols is ultimately due to their reliance on data aggregation as the sole means to achieve privacy, and thus that the ease of isolating nodes from aggregate values results in a failure to protect privacy adequately.

In response, we have proposed a countermeasure against the attack based on data perturbation and optimized with techniques from differential privacy. This approach allows for nodes to protect themselves against the set difference attack by trading accuracy of results against privacy. As we have demonstrated in Section 8.2, this tradeoff is reasonable for realistic scenarios, with the loss of accuracy decreasing quickly as the size of the network increases.

There are still a number of significant avenues to be explored in relation to this work. We have largely avoided an involved mathematical analysis of the feasibility of the set difference attacks in realistic networks, relying instead on simulation. There are many factors that can affect the feasibility of the attack in different networks, and a more rigorous mathematical analysis would be of great use in exploring these and considering approaches towards protecting networks.

The tradeoff between privacy guarantees and the accuracy of results is key to this approach. The use of the differential privacy guarantee provides a well-defined mathematical framework for this tradeoff, and allows for the security parameter to be reduced directly in favour of accuracy. Despite this, the application of the differential privacy guarantee in a wireless sensor network raises a number of issues related to distributed noise generation that we intend to explore in future work. The full implications of combining noise as we have described also remain to be investigated.

An area of great interest in data perturbation for privacy is in how strong privacy guarantees can be maintained over time series data, or highly-linked data-sets. The differential privacy guarantee is extremely strong, but is quickly violated through repeated queries of the same database. When a query can potentially cover a series of readings, the preservation of privacy without adding unacceptably high levels of noise remains open despite some initial results in this area [6, 14].

Our focus in this paper has been exclusively on data aggregation in its simplest form. In some networks, however, there may be a requirement for more complex queries to be distributed across the nodes in the network. The applications of set difference attacks, and the related perturbation defence, to more complex scenarios is worthy of attention.

Finally, the set difference attacks themselves can be extended to consider changes in the network over time. Nodes can join or leave the network, or be included in or excluded from a given aggregate. Nodes may also have predictable data patterns that can be exploited to discount their participation in a given aggregate. These last approaches, which extend the set difference attack to a far wider range of scenarios, is an avenue of great interest in extending the work presented here.

References

1. Denning, D.E., Denning, P.J., Schwartz, M.D.: The Tracker: A Threat to Statistical Database Security. ACM Transactions on Database Systems 4(1), 76–96 (1979)
2. Dwork, C.: Differential Privacy. In: Bugliesi, M., Preneel, B., Sassone, V., Wegener, I. (eds.) ICALP 2006. LNCS, vol. 4052, pp. 1–12. Springer, Heidelberg (2006)
3. Dwork, C., McSherry, F., Nissim, K., Smith, A.: Calibrating Noise to Sensitivity in Private Data Analysis. In: Halevi, S., Rabin, T. (eds.) TCC 2006. LNCS, vol. 3876, pp. 265–284. Springer, Heidelberg (2006)
4. Erdős, P., Rényi, A.: On a classical problem of probability theory. Magyar Tud. Akad. Mat. Kutató Int. Közl 6, 215–220 (1961)
5. European Commission: 95/46/EC-Data Protection Directive. Official Journal of the European Communities 281, 0031–0050 (1995)
6. Ganti, R.K., Pham, N., Tsai, Y.E., Abdelzaher, T.F.: PoolView: stream privacy for grassroots participatory sensing. In: Proceedings of the 6th ACM Conference on Embedded Network Sensor Systems, pp. 281–294. ACM (2008)
7. Gentry, C.: A fully homomorphic encryption scheme. Ph.D. thesis, Stanford University (2009)
8. Gülcü, C., Tsudik, G.: Mixing email with BABEL. In: Symposium on Network and Distributed System Security, pp. 2–16 (1996)
9. He, W., Liu, X., Nguyen, H., Nahrstedt, K., Abdelzaher, T.: PDA: Privacy-Preserving Data Aggregation in Wireless Sensor Networks. In: IEEE INFOCOM 2007 - 26th IEEE International Conference on Computer Communications, pp. 2045–2053. IEEE (May 2007)
10. He, W., Nguyen, H., Liuyi, X., Nahrstedt, K., Abdelzaher, T.: iPDA: An Integrity-Protecting Private Data Aggregation Scheme for Wireless Sensor Networks. In: IEEE Military Communications Conference on MILCOM 2008, pp. 1–7. IEEE (November 2008)
11. Huygens, C., Joosen, W.: Federated and shared use of sensor networks through security middleware. In: Sixth International Conference on Information Technology: New Generations, pp. 1005–1011 (2009)

12. Leontiadis, I., Efstratiou, C., Mascolo, C., Crowcroft, J.: Senshare: Transforming sensor networks into multi-application sensing infrastructures. In: 9th European Conference on Wireless Sensor Networks (February 2012)
13. Murdoch, S.J.: Hot or not: Revealing hidden services by their clock skew. In: 13th ACM Conference on Computer and Communications Security, pp. 27–36. ACM Press (2006)
14. Rastogi, V., Nath, S.: Differentially private aggregation of distributed time-series with transformation and encryption. In: Proceedings of the 2010 International Conference on Management of Data, pp. 735–746. ACM (2010)
15. Yick, J., Mukherjee, B., Ghosal, D.: Wireless sensor network survey. Comput. Netw. 52, 2292–2330 (2008)