Tuning On-Air Signatures for Balancing Performance and Confidentiality

Baihua Zheng Wang-Chien Lee† Peng Liu† Dik Lun Lee‡ Xuhua Ding
Singapore Management University, {bhzheng, xhding}@smu.edu.sg
† The Pennsylvania State University, wlee@cse.psu.edu, pliu@ist.psu.edu
‡ Hong Kong University of Science and Technology, dlee@cs.ust.hk

***Abstract*— In this paper, we investigate the tradeoff between performance and confidentiality in signature-based air indexing schemes for wireless data broadcast. Two metrics, namely *false drop probability* and *false guess probability*, are defined to quantify the filtering efficiency and confidentiality loss of a signature scheme. Our analysis reveals that false drop probability and false guess probability share a similar trend as the tuning parameters of a signature scheme change and it is impossible to achieve a low false drop probability and a high false guess probability simultaneously. In order to balance the performance and confidentiality, we perform an analysis to provide a guidance for parameter settings of the signature schemes to meet different system requirements. In addition, we propose the *jump pointer* technique and the *XOR* signature scheme to further improve the performance and confidentiality. A comprehensive simulation has been conducted to validate our findings.**

I. Introduction

For years, we have envisaged a vision in which the use of high-speed *wireless* devices will facilitate users to watch videos, share pictures, held remote meetings, socialize with friends, and perform many tasks from anywhere at anytime. This vision has come very close to a reality. Exciting news include Intel's plans to embed WiMAX-enabling chips in laptops by the end of 2008, AT&T provides 'All-Access Pass' to the 2008 Olympic Games, and Google TiSP enables FREE in-home wireless broadband service in U.S. and Canada while developing a mobile feature called TiSP on the Run (TiSPOTR) which can provide free wireless broadband service even when the customers are away from the home.

Today, there are many wireless technologies (e.g., Bluetooth, IEEE 802.11, UMTS, Satellite, etc) that could be integrated to construct a seamless, pervasive information access platform. Logically, information access via these wireless technologies can be classified into two basic approaches: *on-demand access* and *periodic broadcast*. On-demand access employs a pull-based approach where a mobile client initiates a query to the server which in turn processes the query and returns the result to the client over a point-to-point channel. On-demand access is suitable for lightly loaded systems in which server processing capacity and wireless channels are not severely contended.

On the other hand, periodic broadcast requires the server to pro-actively push data to the clients over a dedicated broadcast channel. This approach allows an arbitrary number of clients to access data simultaneously, and thus is particularly suitable for heavily loaded systems. Wireless data broadcast services have been available as commercial products for many years, e.g. StarBand (www.starband.com) and Hughes Network (go.gethughesnet.com). Recently, there has been a push for such systems from the industry and various standard bodies. For example, born out of the International Telecommunication Union's (ITU) International Mobile Telecommunications "IMT-2000" initiative, the Third Generation Partnership Project 2 (www.3gpp2.org) is developing Broadcast and Multicast Service in cdma2000 Wireless IP network.

In a wireless data broadcast environment, clients with appropriate equipments can monitor the broadcast channel and log the data items being broadcast. As a result, if the broadcast data items are not encrypted, their content is open to the public (i.e., any person can access it for free). Key-based encryption is a natural choice for ensuring secure access of data on air (i.e., only subscribers who own valid keys can decrypt the received packets to access the data items). However, a search for broadcast data items needs to first receive all the broadcast data items off the air for decryption and further processing (i.e., filtering out unwanted data items). In other words, no matter how many data items a client requests, she has to download and decrypt all the data items, which consumes energy significantly.

To help alleviate the high cost of receiving, decrypting and filtering broadcast data, auxiliary information can be provided on the broadcast channel to annotate the broadcast data items. This technique is called *air indexing*. The basic idea is that, based on index information broadcast along with data items (including indexed attribute values, arrival schedule, length of data items, etc.), mobile clients can skip the retrieval of unwanted data items via tuning into *doze mode* and switch back to *active mode* only when the data of desire arrives. Existing air indexing techniques mainly focus on the search performance issues, i.e., designing index structures and corresponding search algorithms to enable fast data retrieval.

Differently, the work presented in this paper tries to address both *performance* and *confidentiality* issues of wireless data broadcast services. We propose to use signature-based air index which is a typical *approximate* index. It uses a fixed length bit-vector (i.e., a signature) to encode all the indexed attributes and clients can check whether a data item satisfies the query by only checking its corresponding signature. However, a signature is an abstract of the real attribute values and different items may share the same signature. A match between a signature and the query does not guarantee that the real data item actually satisfies the query. Consequently, a signature keeps some uncertainty of the indexed data item that may deteriorate the search performance. From aspect of confidentiality, the approximate nature of the signature is desirable as it actually hides partial information from the public. The index information (i.e., signature), even broadcast

in a non-encrypted form, prevents unauthorized attackers to infer data content on broadcast. This work studies the tradeoff between performance and confidentiality of different signature settings, by analysis and experimentation. In addition, we take a step further to enhance the performance and confidentiality of signature-based index by incorporating the jump pointer technique and the XOR operation. The main contributions of our study are five-fold.

- The issue of confidentiality loss in air indexing is, to the best knowledge of the authors, identified and studied for the first time in this research work[1].
- The tradeoff between performance and confidentiality loss in signature-based air indexes is analyzed in terms of false drop and false guess probabilities of the signatures.
- An analytical model is developed to analyze the impact of different control parameters, which serves as a guidance for configuring signatures to meet the performance and confidentiality requirements of wireless data broadcast applications.
- The jump pointer technique and the XOR signature scheme have been proposed to further improve the performance and confidentiality of signature-based air indexes, respectively.
- Extensive experiments (based on both simulations and prototyping on PDA) are conducted to validate our analysis and to evaluate the examined signature schemes.

The rest of this paper is organized as follows. Section II and Section III present the preliminaries and related work of this study, respectively. Section IV formulates the problem, and defines/analyzes the metrics for performance and confidentiality, i.e., the false drop probability as well as the false guess probability. Section V derives an analytical model that balances performance and confidentiality in a secure wireless broadcast system, and proposes the jump pointer technique and the XOR signature scheme, two strategies to further improve the performance and confidentiality of a signature-based system, respectively. Section VI reports the simulation results to validate our analysis. Finally, Section VII concludes this paper.

II. Preliminaries

In this section, we first give an overview of the signature technique and its implementation in a wireless broadcast system, and then describe the adopted system model and explain all the assumptions we make.

A. Overview of the Signature Techniques

Data Item:	*Attr. 1: Security*	*Attr. 2: Pervasive*
Security	*001 100 001 001*	
Pervasive	∨)	*101 000 100 001*
Data Signature	S_i	*101 100 101 001*

(a) Signature generation

Query Q	*Query Signature* S_Q	$S_Q \wedge S_i$	*Results*
Hacker	*000 101 000 101*	*000 100 000 001*	*No Match*
Security	*001 100 001 001*	*001 100 001 001*	*True Match*
Mobile	*100 100 001 001*	*100 100 001 001*	*False Match*

(b) Query comparision

Fig. 1. Signature generation and comparison

[1]A preliminary report of this study appeared in Proc. the ACM 14-th Conference on Information and Knowledge Management (CIKM'05) [23].

Signature techniques have been studied extensively in information retrieval [15], [18]. Different from the term *digital signature* used in the security context, a signature in the database context is basically an abstract of the information in a data item, which contains a set of attributes. Given a set of data items to be indexed by multiple attributes, the signature S_i of data item i is typically formed by first hashing each indexed attribute in the data item into a *bit string* and then *superimposing* (i.e., bitwise-OR, denoted as $\vee$) all these bit strings into a signature. Note that the size of a signature equals the size of the bit string. An example of signature generation is depicted in Figure 1(a), in which each attribute is hashed into a 12-bit string.

To process a query which contains one searched attribute, a *query signature* S_Q corresponding to the query Q is generated first, based on the same hash function. Thereafter, S_Q is compared against the signatures of examined data items using bitwise-AND (denoted as $\wedge$). The signatures *match* if for every bit set in S_Q, the corresponding bit in the compared data signature S_i is also set. There are two possible outcomes of the comparison:

- $S_Q \wedge S_i \neq S_Q$: data item i does not match query Q.
- $S_Q \wedge S_i = S_Q$: a match has two possible implications:
  - *true match*: the data item is really what the query searches for; and
  - *false drop*: the data item in fact does not satisfy the search criteria although the signature comparison indicates a match.

As shown in Figure 1(b), three queries are issued and their corresponding signatures are produced based on the same hash function. According to the result of $S_Q \wedge S_i$, the examined data item is not qualified for the first query, Q_1=Hacker, but qualified for the other two queries, Q_2=Security and Q_3=Mobile. It is a true match for Q_2 as the data item does contain the attribute Security while it is a false drop for Q_3 because the data item does not contain the queried attribute Mobile.

Bloom Filter, commonly used in networking, shares some similarities as signature [3]. However, it adopts multiple hash functions to generate bit strings. Given a bit string with w_b bits set to 1, it needs w_b hash functions with one corresponding to one bit. As a result, the generation and comparison of the bit strings become more complicated and time-consuming that is not suitable for the wireless broadcast systems.

B. System Model

Like most of the work in the literature, we, in this work, assume a generic wireless data broadcast system that consists of three parts: 1) a broadcast channel[2]; 2) the broadcast server; and 3) mobile clients [25], [27], [28]. Figure 2 shows a high-level overview of the system model. The broadcast server is interfaced with other data sources via high speed networks and thus can be considered as a logical data source for all the mobile users in the system. The server is responsible for generating the broadcast program, i.e., *periodically* encrypting and disseminating data items to its clients via the shared broadcast channel. A complete broadcast of data items is called a *broadcast cycle*. All the items considered in this model consist of a set of searchable attributes and a content body.

[2]The channel is an abstraction for a communication medium and it is used as a generic term to refer to a computer network, satellite channel, TV channel, radio channel or a virtual circuit over a cellular channel [14].

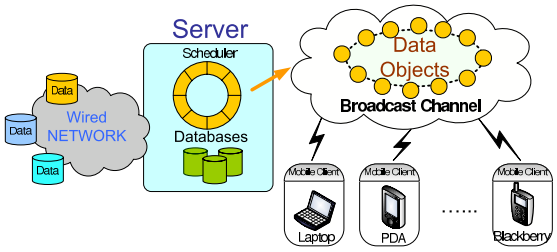


Fig. 2. A wireless data broadcast system.

The mobile clients also play an important role in the system due to the client-side processing. Each client has to continuously monitor the broadcast channel to receive the data items of interests (as specified by user queries). Without any auxiliary information, a client has to continuously listen to the channel for one complete broadcast cycle, comparing the searchable attributes of each item against the query predicates to process a query. Suppose only 5% of the data items are qualified for the query, the retrieval of the other 95% of data items may be redundant.

Signature techniques, well known for the ability to efficiently filter the unqualified data against a query, have been proposed for air indexing [17]. The idea is to put s data items into a group, namely a *data frame*, and generate a signature for each data frame via superimposing s signatures corresponding to s data items. Compared with the data items in a data frame, a signature is usually much smaller and hence the retrieval of a signature is much cheaper than retrieval of the data frame. Consequently, in a wireless broadcast system, the signatures and data frames are interleaved, as shown in Figure 3.

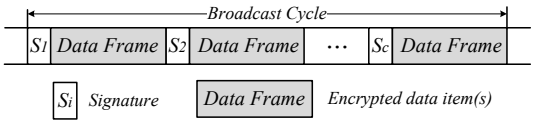


Fig. 3. Signature based air index

We assume that the hash function H adopted to generate the signatures is known to all the clients. Whenever a client issues a query and tunes into the channel, it generates the signature for the query based on H, and waits for the first coming signature. In order to facilitate the initial probing, we assume the data frame and index information are packed into packets in the broadcast channel and each packet contains a pointer to the next signature packet. Consequently, a client knows when to wake up to receive upcoming signatures. Thereafter, it retrieves the signature for comparison. If the signature matches the query, it retrieves the following data frame. Otherwise, it tunes into the *doze* mode and only switches to the *active* mode when the next signature is about to broadcast. The process continues for a complete broadcast cycle.

For example, assume $s = 1$ (i.e., each data frame contains only one data item) and the broadcast system is going to broadcast an item i with security and pervasive as two key attribute values. A client who issues a query Q may first retrieve the signature $S_i = 101100101001$ and compare that with its query signature S_Q to decide whether data item i satisfies the query. For example, a client who issues Q_1 = hacker can safely ignore the downloading of the item as $S_{Q_1} \wedge S_i \neq S_{Q_1}$. Another client who issues Q_2 has to download the item because $S_{Q_2} \wedge S_i = S_{Q_2}$, indicating the item might satisfy the query.

Without loss of generality, we assume the wireless broadcast system adopts various approaches to guarantee the integrity of the broadcast data. The underlying wireless medium is secure but open to the public. Consequently, both authorized users and adversaries can retrieve the packets from the wireless channel. In order to make sure only subscribed users can enjoy the services, all the 'data' packets are encrypted using keys that are known to the subscribed users while the index information (i.e., signatures) are broadcast in a non-encrypted form to facilitate quick search and filtering. We assume the system adopts suitable key management techniques to manage keys.

III. Related Work

In this section, we first review the security issues related to wireless broadcast, and then we briefly review other related work.

A. Secure Broadcast

Wireless data broadcast, an approach for delivering information to a group of users simultaneously, has been adopted in many applications. In this subsection, we review some security issues and solutions related to wireless broadcast in the conventional wireless broadcast scenarios and wireless sensor network (WSN) scenarios, respectively. In the context of wireless broadcast systems, the security problems can be divided roughly into two intertwined areas, i.e., *secrecy* and *authentication*. The former is to guarantee that only the subscribers can receive and recover the broadcast information while the unauthorized users cannot. A simple approach is via information encryption. The latter is to make sure that the received information is from the right source/sender and most approaches rely on asymmetric digital signatures[3].

In a wireless sensor network, each sensor node communicates with its neighbors via wireless broadcast. It shares similar security concerns as traditional wireless broadcast systems, i.e., the confidentiality of the transferred data should be guaranteed, and the authentication of the source should be enabled. However, different from wireless broadcast systems where the connection between a user and the wireless channel is not affected by other users, each sensor node contributes to the network topology which might be dynamic. In addition, the sensor nodes have limited processing power and energy which can be easily exhausted by intensive computations. All these constraints introduce new challenges [4].

Symmetric key cryptography is the most feasible encryption mechanism for node to node communication, and key predistribution is a popular key management approach in WSN [5], [6]. The main idea is to pre-install a limited number of keys in sensor nodes prior to actual deployment. In the context of WSN, it has to ensure good network connectivity through key sharing and resilience to node/key captured by the enemy as well. Some other related work includes TinySec [19] and μTESLA [21]. The former develops mechanisms for link-level encryption, and the latter is a protocol to provide authenticated broadcast for severely resource-constrained environments.

Although the security issue has been studied in traditional wireless broadcast systems as well as wireless sensor networks, it is worth to note that all the existing approaches rely purely

[3]Note that while the term *digital signature* is similar to *data signature* discussed in Figure 1, they are fundamentally different. This paper focuses on data signature techniques.

on the cryptography techniques. Their main focus is to design different cryptography related techniques against various attacks. However, the focus of this work is to facilitate *efficient search* while ensuring *high confidentiality* by configuring signature-based air index. To the best of our knowledge, this is the first work that aims at integrating confidentiality protection with filtering/search function using indexing techniques. As a data signature uses a bit vector of fixed length to represent a set of attributes, uncertainty between true matches and false drops is introduced. Uncertainty may deteriorate the filtering efficiency of a data signature scheme. However, it also offers an opportunity to encode information. In the literature, all the existing work related to signature-based indexing aims at reducing the false drop in order to improve the filtering efficiency. In this paper, we analyze how to protect confidentiality via increasing false drops and strike a balance between performance and confidentiality.

B. Other Related Work

Wireless broadcast is a popular approach for disseminating commonly interested information to clients to enable a simultaneous access [1]. Air indexing is widely adopted to conserve battery power in mobile clients. Several tree-based indexing techniques, such as flexible indexing and distributed tree indexing, for broadcast channels are proposed [12], [13]. However, these studies focus on one-dimensional indexes for equality-based queries. The processing of general queries with a semantics-based broadcast approach is proposed [16], and the range query is studied in [22]. Traditional index techniques, such as hashing [13] and signature file [9], are also applied in air indexing, along with hybrid approach [10]. Besides the design of different indexing structures for different scenarios, index organization algorithms are also studied [14]. However, none of the above techniques addresses any security issue.

On the other hand, security is a major concern in various services and applications. The issues of protecting user privacy due to insufficient anonymity are addressed by introducing dummy users and controlling the update frequency/spatial resolution in [2]. In [7], [8], a client-trusted middleware architecture is assumed to transform requests from mobile clients into new queries. In these work, a request based on a user position is masked as a region request which will only be issued if there are at least k requests from the same region. Thus, the location service provider cannot deduce individual client position.

Security is also an important topic in mobile ad-hoc networks (MANET). Due to the freedom of nodes to join, leave, and move inside the network, MANET is vulnerable. Consequently, different techniques have been proposed to protect against various attacks [4]. For example, *Watchdog* and *Pathrater* have been proposed to mitigate the routing mis-behaviors in MANET [20], and a cluster-based intrusion detection technique for ad-hoc network has been proposed [11]. However, key management and cryptography are not the focus of this paper. We address the confidentiality issue from data management aspect.

IV. Problem Formulation

Our study aims at revealing important and practical insights on design, deployment and administration of signature techniques in wireless broadcast systems in order to address both performance and data confidentiality concerns. We assume that the system maintains a combined domain of indexed attributes, denoted as D_A, for searching data on air. In other words, data items maintained in the server are indexed by attributes in D_A to facilitate efficient search. In order to construct a broadcast program, the *system administrator* chooses a hash function H for signature generation, decides the number of attributes to be indexed in signatures, and groups data items into data frames. Our goal is *to obtain a signature configuration which minimizes confidentiality loss while not causing much performance deterioration*. As the foundation of this work, we, in this section, define the performance and confidentiality metrics, and analyze the factors that affect those metrics. Table I summarizes the notations used in our analysis.

TABLE I
SUMMARY OF SYMBOLS IN ANALYSIS

A	an application;
D_A	the combined domain of indexed attributes in A;
D_H	the hacker's dictionary;
N	number of the data items included in a broadcast cycle;
C	number of signatures in a broadcast cycle;
C_f	number of matched signature due to false drops;
C_t	number of matched signature due to true matches;
$C_{m'}$	number of signatures that do not match;
G	number of total guesses based on a hacker's dictionary;
G_f	number of false guesses;
G_t	number of correct guesses;
m	number of bits in a signature;
n	the size of the content body for each item;
u	number of attributes indexed in a data item;
s	number of data items included in a data frame;
w_b	number of 1's in an attribute's signature;
w_f	average number of 1's in a data item's signature;
P_f	false drop probability;
P_g	false guess probability;

A. Performance Metrics

Access time and *tuning time* have been widely used as performance metrics in the studies of wireless data broadcast [14]. The former represents the access latency and the latter estimates the energy consumption in mobile clients. The index information can save the retrieval of unwanted data items and hence it can improve the tuning time performance. However, the index information also consumes extra bandwidth which as a result extends the broadcast cycle. As the result set to a query might contain multiple data items, the client has to listen to the entire broadcast cycle to avoid any miss of the right answer. Hence, the access time is only affected by the bandwidth overhead incurred due to the signature. From the access time point of view, the size of each signature is preferred to be small. On the other hand, the tuning time performance depends on the filtering ability of signatures. As long as a signature shows a match, whether a true match or a false drop, the data items have to be downloaded and decrypted for further checking. Therefore, the false drop probability should be reduced to improve the tuning time performance and hence the energy consumption.

To be more specific, the access time, denoted as ACC, and the tuning time, denoted as $TUNE$, are expressed in Equation (1) and Equation (2), respectively. As we assume that the query process is finished only when all the answer items are retrieved, a client has to download all the signatures within one cycle and the access time for a given query is one broadcast cycle, denoted as $CYCLE$. In addition, the client needs to perform an initial probe before a signature is received, denoted by $PROBE$, which

on average is a half of the summation of a data frame and its signature. On the other hand, the tuning time is the time a client stays active for the initial probe $PROBE$, scans all the signatures within one cycle denoted by SIG, and retrieves all the matched frames (including both true matches and false drops).

$$\begin{aligned} ACC &= PROBE + CYCLE \quad (1)\\ &= \frac{m + n \cdot s}{2} + C \cdot (m + n \cdot s) \\ TUNE &= PROBE + SIG + C_t \cdot n \cdot s + C_f \cdot n \cdot s \quad (2)\\ &= \frac{m + n \cdot s}{2} + C \cdot m + n \cdot s \cdot (C_t + P_f \cdot (C - C_t)) \end{aligned}$$

Obviously, false drop probability P_f has a direct impact on the tuning time performance. In order to have a better understanding of those factors that might affect P_f, we conduct an analysis of P_f. Semantically, P_f refers to the probability that the signature of a data item matches the query signature, yet the data item actually does not satisfy the query. Given a query Q, let $C_{m'}$ be the number of signatures that do not match S_Q, C_t be the true matches, and C_f be the false drops with $C = C_t + C_f + C_{m'}$. False drop probability P_f is defined as the ratio of C_f to $(C - C_t)$.

$$P_f = \frac{C_f}{C - C_t} \quad (3)$$

Since both the hash collision and superimposition of bit strings in signature generation may cause false drops, we use $P_{f,col}$ and $P_{f,sup}$ to denote the false drop probabilities caused by hash collision and superimposition, respectively. In order to analyze hash collision, a *collision factor* is used to denote the average number of different inputs hashed into the same output. For an application A, let $CF_{A,sig}$ denote the average number of data items hashed into the same signature. $P_{f,col}$ and $P_{f,sup}$ can be derived based on Equation (4) and Equation (5), respectively. Please refer to [23] for detailed derivation. Here, $\mathsf{comb}(\cdot,\cdot)$ represents the binomial function.

Based on the above analysis, we obtain the following observations. First, given the fact that $|D_A|$ is limited, it is very likely that $CF_{A,sig}$ is very close, if not equivalent to, 1. Take a common setting as an example. With $m = 64$ and $w_b = 4$, a total number of $\mathsf{comb}(64, 4) = 635,376$ attributes can be indexed without causing a hash conflict. Consequently, $P_{f,sup}$ plays a very important role in affecting P_f and hence the tuning time performance. Second, P_f is closely related to the following factors: 1) the signature length m, 2) the bit setting w_b, 3) the number of indexed attributes superimposed into a data signature u, and 4) the number of items in an integrated group s. Therefore, by tuning these parameters, the system can adjust P_f. More specific, P_f is increased if we decrease m and/or increase u and/or s. The tuning of w_b is more complicated, which will be explored by our simulation in Section VI.

$$\begin{aligned} P_{f,col(D_A)} &= \frac{C_f}{C - C_t} = \frac{C_t \cdot (CF_{A,sig} - 1)}{\mathsf{comb}(I_f, s)} \\ &\approx \Big((\frac{|D_A|}{|D_A| - u})^s - 1\Big) \cdot \Big(CF_{A,sig} - 1\Big) \quad (4)\\ P_{f,sup} &= \frac{\mathsf{comb}(w_f, w_b)}{\mathsf{comb}(m, w_b)} \approx (1 - e^{-\frac{w_b \cdot u \cdot s}{m}})^{w_b} \quad (5) \end{aligned}$$

B. Confidentiality Metric

From the confidentiality aspect, the fact that one signature can match different queries introduces an uncertainty which actually prevents attackers (also called *hackers*) from easily knowing the indexed attribute values of data items. The hackers scan the broadcast channel, download indexes and data items, and try to guess the encrypted content of data items from indexing information. Hence, when a hacker downloads a signature from the broadcast channel, he might start a *dictionary attack* by using all the attribute values in his dictionary D_H to generate $|D_H|$ signatures and compare each of them with a downloaded signature. Assuming that the attacker's dictionary is comprehensive (i.e., $D_A \subseteq D_H$), she will find a set of matches in D_H. Among those matches, there are *correct guesses* and *false guesses*. In order to quantify how much information has been leaked to attackers, *information leaking degree (ILD)* is introduced as the confidentiality metric in this paper. As defined in Equation (7), ILD is the ratio of the number of correct guesses G_t to the total number of matched guesses G (= $G_t + G_f$).

Given a signature corresponding to $u \cdot s$ attributes, the number of correct guesses is fixed (i.e., $u \cdot s$) and that of false guesses G_f depends on the *false guess probability* P_g. It is defined as the probability that a dictionary value matches a signature of a data frame but actually none of the data items included in the data frame matches the dictionary value, as expressed in Equation (6).

$$\begin{aligned} P_g &= \frac{G_f}{|D_H| - G_t} \quad (6)\\ ILD &= \frac{G_t}{G_t + G_f} = \frac{G_t}{G_t + P_g \cdot (|D_H| - G_t)} \quad (7) \end{aligned}$$

Similar to the false drop probability P_f, P_g is also affected by both the hash collision and superimposition of bit strings in signature generation. Let CF_H denote the collision factor associated with D_H (usually much larger than D_A), G_f can be approximated by $(CF_H - 1) \cdot G_t$. Therefore, false guess probability caused by hash collision $P_{g,col(D_H)}$ is defined in Equation (8) and that caused by superimposition $P_{g,sup}$ is defined in Equation (9).

$$\begin{aligned} P_{g,col(D_H)} &= \frac{G_t \cdot (CF_H - 1)}{|D_H| - G_t} \quad (8)\\ P_{g,sup} &= P_{f,sup} \quad (9) \end{aligned}$$

The above analysis reveals some interesting findings. When $P_g = 0$, the information leaking degree ILD is 100%. To reduce ILD to 50%, P_g needs to be raised up to $\frac{G_t}{|D_H| - G_t}$. This finding indicates a dependency between information leaking degree and the false guess probability. In addition, it points out an observation, i.e., having a reasonable false drop probability, which behaves similarly as the false guess probability, is not such a bad idea for the sake of confidentiality concerns, even though low false drop probability is preferred from the performance point of views.

V. System Tuning

This study attempts to correlate performance and confidentiality requirements of the signature-based air indexing techniques for wireless broadcast systems and investigate the tradeoff between them. Ideally, a secure wireless data broadcast system should provide an efficient data access with minimal information leakage. In other words, it requires the false drop probability P_f to be small but false guess probability P_g to be large. However, based on the analytical study conducted in previous section, we understand both P_f and P_g share the same trend and it is

impossible to optimize both security and energy at the same time. In order to balance the performance and confidentiality in the signature-based system, we, in this section, first develop a cost model to guide the signature configurations for various system requirements. Thereafter, we present *jump pointer* technique and *XOR* signature, two approaches that can improve the performance and confidentiality, respectively.

A. Balance Performance with Confidentiality

Based on the analysis, we understand that there are four parameters that can affect the performance and confidentiality, including i) the number of bits in one signature m; ii) the fixed number of bits set to 1 in a bit string w_b; iii) the number of attributes in a data item that contribute to the signature u; and iv) the number of items that contribute to the signature s.

In many cases, the settings of m and u are relatively stable. In traditional information retrieval applications, the size of the signature, m, is set to a large value and the number of bits set, w_b, is carefully selected to provide a large space of hashed bit strings and minimize hash collisions. However, for secure wireless data broadcast systems, a long signature consumes too much bandwidth and extends both access latency and tuning time. Furthermore, a long signature may result in a higher *ILD*. Consequently, the value of m usually is kept in a reasonable size (e.g., 64 - 256 bits), and u is normally application-dependent. On the other hand, the settings of w_b and s are much more flexible. The former can be tuned from 1 to m and the latter can be set to 1, 2, etc. In the following, we analyze how to tune these parameters in order to optimize the overall system cost that considers both confidentiality metric and performance metric.

We take an encrypted broadcast system which does not provide any index information as a reference system. A client has to retrieve all the packets in such an encrypted broadcast system. Consequently, both tuning time and access time, defined as T_{ref} and A_{ref} respectively, equal a complete broadcast cycle, as defined in Equation (10). When the index is provided, the client may save the retrieval of some unnecessary data frames and hence the tuning time performance is improved. Nevertheless, the index information occupies extra bandwidth which extends the access time. In order to quantify the performance gain, Equation (11) is defined. Here, the tuning time/access time under the reference system are used as the baseline performance, and the parameter δ, ranging over $[0, 1]$, is used to adjust the importance between the tuning time performance and access time performance. On the other hand, a plain index may release some information of the encrypted data and hence cause information leakage. Taking both the performance gain and confidentiality loss into consideration, the normalized system cost is defined in Equation (12). The first term C_{per} stands for the performance, and the second term ILD is the confidentiality metric. The parameter α, ranging over $[0, 1]$, assigns different weights to performance and confidentiality. The larger α is, the more important the performance is considered. It is noticed that the cost under the reference system is zero.

$$T_{ref} = A_{ref} = \frac{n}{2} + N \cdot n \quad (10)$$

$$C_{per} = \delta \cdot \frac{TUNE - T_{ref}}{T_{ref}} + (1-\delta) \cdot \frac{ACC - A_{ref}}{A_{ref}} \quad (11)$$

$$C_{nor} = \alpha \cdot C_{per} + (1-\alpha) \cdot ILD \quad (12)$$

Based on Equation (1), only the parameter s, but not w_b, has an impact on the access time performance. It is observed that as s increases, the probe time (i.e., $PROBE$) gets extended while the broadcast cycle (i.e., $CYCLE$) is shorten. Given two settings of s, denoted as s_1 and s_2, the difference of the corresponding access time is

$$\triangle = ACC_{s_1} - ACC_{s_2} = (s_1 - s_2) \cdot (\frac{n}{2} - \frac{m \cdot N}{s_1 \cdot s_2}) \quad (13)$$

Without loss of generality, we assume $s_1 = s + 1$ and $s_2 = s$ and Equation (13) can be reexpressed as shown in Equation (14). Initially, $\triangle$ is negative which means an increase of s reduces the access time. This is because the benefit of reduced broadcast cycle pays off the cost of extended probe time. However, as $s \cdot (s+1)$ value increases, $\triangle$ becomes positive which means a further increase of s actually extends the access time. Consequently, the best access time performance can be achieved when $\triangle$ is about to change its value from negative to positive, denoted as s_{acc}. According to quadratic formula, s_{acc} can be derived as in Equation (15).

$$\triangle = \frac{s \cdot (s+1) \cdot n - 2 \cdot m \cdot N}{2 \cdot s \cdot (s+1)} \quad (14)$$

$$s_{acc} = \frac{\sqrt{n^2 - 8 \cdot n \cdot m \cdot N} - n}{2 \cdot n} \quad (15)$$

However, the setting of s also affects false drop probability P_f. As s increases, P_f increases as well. When s becomes too large, it is very likely that all the bits of a signature are set to 1. Hence, the signature loses its filtering capability and it will anyway indicate a match for any query. In other words, the signature with all the bits set to 1 forces the user to retrieve each single data item, no matter whether it satisfies the query or not. Consequently, the value of s should be carefully selected. In order to simplify our discussions, we assume the possible values of s vary in a small range (e.g., $\{1, 2, 4\}$).

Next, we analyze the impact of the parameter w_b on the system cost. The system cost under fixed u, m and s is defined in Equation (16). As the first term, i.e., $\alpha \cdot \delta \cdot \frac{SIG}{T_{ref}}$, and the third term, i.e., $\alpha \cdot (1-\delta) \cdot \frac{ACC - A_{ref}}{A_{ref}}$, are constant under fixed u, m and s, C_{nor} with various w_b is affected by $\alpha \cdot \delta \cdot \frac{C'_m \cdot n \cdot s}{T_{ref}}$ and $(1-\alpha) \cdot ILD$. In other words, C_{nor} is minimized when $(1-\alpha) \cdot ILD - \alpha \cdot \delta \cdot \frac{C'_m \cdot n \cdot s}{T_{ref}}$ (denoted as θ) is minimized.

$$C_{nor} = \alpha \cdot \delta \cdot \frac{SIG}{T_{ref}} - \alpha \cdot \delta \cdot \frac{C'_m \cdot n \cdot s}{T_{ref}} + \alpha \cdot (1-\delta) \cdot \frac{ACC - A_{ref}}{A_{ref}} + (1-\alpha) \cdot ILD \quad (16)$$

$$\theta = (1-\alpha) \cdot ILD - \alpha \cdot \delta \cdot \frac{C'_m \cdot n \cdot s}{T_{ref}} \quad (17)$$
$$= \frac{(1-\alpha) \cdot G_t}{G_t + P_g \cdot (|D_H| - G_t)} + \frac{\alpha \cdot \delta \cdot n \cdot s}{T_{ref}} \cdot P_f \cdot (C - C_t) - \frac{\alpha \cdot \delta \cdot n \cdot s \cdot (C - C_t)}{T_{ref}}$$

As shown in Equation (17), θ is affected by false drop probability P_f and false guess probability P_g. Recall that, as our analysis in Section IV shows, P_f and P_g are correlated, with $P_{f,sup} = P_{g,sup}$. Without loss of generality, we assume that a proper signature generating function usually offers a near-zero

hash collision, i.e., $P_{f,col} \approx 0$ and $P_{g,col} \approx 0$. For example, a hash function with $m = 64$ and $w_b = 4$ can generate $\mathsf{comb}(64, 4) \approx 6.4 \times 10^5$ unique signatures and that with $m = 128$ and $w_b = 4$ can generate $\mathsf{comb}(128, 4) \approx 10^7$ unique signatures. Consequently, we assume when the normalized cost is minimized, $P_f \approx P_g = p$ and hence we can derive the value of p (i.e., P_f and P_g) according to quadratic formula, as derived in Equation (18).

$$\begin{aligned} p &= \frac{\sqrt{b^2 - 4 \cdot a \cdot c} - b}{2 \cdot a} \\ \text{where } a &= \alpha \cdot \delta \cdot n \cdot s \cdot (C - C_t) \cdot (|D_H| - G_t) \\ b &= \alpha \cdot \delta \cdot n \cdot s \cdot (C - C_t) \cdot G_t \\ c &= -(1 - \alpha) \cdot G_t \cdot T_{ref} \end{aligned} \tag{18}$$

If a range of s, denoted as R_s, is given, we can derive the optimal cost C_{nor} under each $s \in R_s$, denoted as $C_{nor}(s)$. The best setting of s is set to the one s' with minimal cost, i.e., $\forall s \in R_s$, $C_{nor}(s) \geq C_{nor}(s')$.

B. Jump Pointer Technique

Although the signature-based approach can skip the download of some unnecessary data items by carefully tuning the false drop probability P_f, it still requires the mobile clients to listen to *all* the signatures in order to avoid any false match. Take a broadcast cycle containing 10000 data items with $s = 1$ as an example. Suppose only 5% of the data items satisfy the query and the estimated false drop probability P_f is 0.5, the number of signatures that do not match the query $C_{m'}$ is 4750. In other words, among 10000 signatures, 4750 signatures do not match the query and hence the retrieval and comparison of those signatures do not contribute to the query result.

In addition, the value of $C_{m'}$ also affects the switching cost. Air indexing techniques aim at keeping the mobile clients in doze mode as long as possible and only switching them back to active mode when the data of interest are broadcast. Although most, if not all, of the existing works in the literature neglect the energy overhead incurred by mode switching, it is observed that the typical setup time for a mobile device to start or tune into active mode is, while device-dependent, usually in the order of 100 μs [24]. Consequently, the switching cost in terms of energy consumption is not negligible. As $C_{m'}$ increases, the number of switches goes up as well. Therefore, it is desirable to reduce $C_{m'}$ value in order to reduce the tuning time and switching cost and hence the energy consumption.

$C_{m'}$ $(= C - C_t - C_f)$ can be reduced if C_f is increased. Consequently, a straightforward approach is to introduce more false drops as $C_f = P_f \cdot (C - C_t)$. However, this approach is not appealing as each false drop requires the client to retrieve and to decrypt corresponding data frame that significantly extends the tuning time. Thus, we propose a simple but novel approach to tackle this issue, namely the *jump pointer* technique. The basic idea is that when the broadcast server schedules the data items for broadcast, it has the full knowledge of the indexed attributes for each data frame and hence it knows whether two data frames can satisfy the same query. Given two frames f_1 and f_2, we assume their corresponding signatures are s_1 and s_2, and the corresponding attribute lists are A_{s_1} and A_{s_2}. If $A_{s_1} \cap A_{s_2} = \emptyset$, f_2 will not satisfy any query that f_1 satisfies. For a given query Q, if f_1 truly matches it, the retrieval of f_2 can be safely ignored. Based on this knowledge, it associates a pointer, namely *jump pointer* p_j, to each signature s, pointing to the next (closest) signature s' in the broadcast cycle that might satisfy the same query as s, i.e., $A_s \cap A_{s'} \neq \emptyset$.

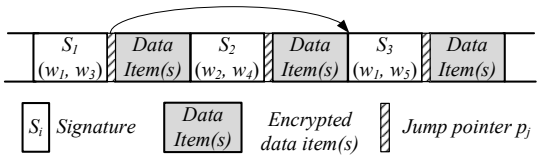


Fig. 4. Example jump pointer

Figure 4 depicts an example, in which the jump pointer associated with the signature s_1 points to s_3, as $A_{s_1} \cap A_{s_2} = \emptyset$ and $A_{s_1} \cap A_{s_3} = \{w_1\}$. Here, w_i represents an indexed attribute value and the set of w_i inside the parenthesis below each signature s_j represents A_{s_j}. For example, the signature s_1 is generated based on w_1 and w_3, and the signature s_2 is generated by w_2 and w_4. Suppose a client issues a query $q = \{w_1\}$, and s_1 is the first complete signature it receives. As s_1 matches the query signature s_q, it downloads the data frame and finds a true match. Consequently, it can skip the signature s_2 and its data frame (in doze mode) by following the jump pointer p_j to access s_3. The jump pointer technique does not cause any false miss, as proved by Lemma 1.

***Lemma** 1:* The jump pointer technique is free of false misses. Suppose a query q on attribute a_q finds a true match at signature S_i, and it follows the jump pointer to access signature S_j. All the data frames broadcast between S_i and S_j do not contribute to the result set of q. □

Proof. Suppose that there is a data item d (associated with a signature S_d) broadcast between S_i and S_j satisfies the query q, i.e., item d has a_q as one of its indexed attribute values. Consequently, S_d must have its indexed attributes list A_{s_d} include a_q, i.e., $a_d \in A_{S_d}$. As the query q finds a true match at signature s_i, the indexed keywords list A_{S_i} contains a_q as well. As a result, $a_q \in (A_{S_i} \cap A_{S_d})$ and $A_{S_i} \cap A_{S_d} \neq \emptyset$. Consequently, the jump pointer with s_i will not point to S_j, which contradicts our assumption. ■

Let $|D_A|$ be the size of the combined domain of indexed attributes, s be the number of data items included into one data frame, and u be the number of attributes indexed for each data item, the probability ρ that two given data frames might satisfy the same query can be approximated by Equation (19). For example, when $|D_A| = 1000$, $u = 10$, and $s = 1$, $\rho = 0.092$ which means a jump pointer on average points to the signature that is $\frac{1}{\rho} = 10.4$ data frames apart away from the current one. Given a dataset containing C data items, one broadcast cycle has $\frac{C}{s}$ data frames with each having $\frac{s \cdot u}{|D_A|}$ probability to match a specific query. Consequently, the jump pointer technique can save roughly $C_s = \frac{C}{s} \times \frac{s \cdot u}{|D_A|} \times \frac{1}{\rho}$ signature retrievals. Among those C_s saved signature retrievals, some might match the query while the others do not. Consequently, C_s affects both false drops C_f and the number of signatures that do not match the query $C_{m'}$. Based on the assumption that the ratio of C_f to $C_{m'}$ does not change, the false drop probability remains, as proved by Lemma 2.

$$\rho = 1 - \frac{\mathsf{comb}(|D_A| - s \cdot u, s \cdot u)}{\mathsf{comb}(|D_A|, s \cdot u)} \tag{19}$$

***Lemma** 2:* The jump index technique reduces the total number of index signatures checked, but it does not change the false drop probability. □

Proof. Without the jump pointer, $P_f = \frac{C_f}{C-C_t}$. If we assume $\frac{C_f}{C_{m'}} = a$, $P_f = \frac{C_f}{C_f+C_{m'}} = \frac{a}{a+1}$. When the jump pointer is provided, C_t remains but both C_f and $C_{m'}$ are reduced to C'_f and $C'_{m'}$. Let $\triangle_1 = C_f - C'_f$, $\triangle_2 = C_{m'} - C'_{m'}$, and $C_s = \triangle_1 + \triangle_2$ (i.e., the number of total signature checks saved). Assume $\frac{C'_f}{C'_{m'}}$ remains a, the new false drop probability P'_f can be derived as follows.

$$P'_f = \frac{C_f - \triangle_1}{C_f - \triangle_1 + C_{m'} - \triangle_2} = \frac{a \cdot C_{m'} - \frac{a}{a+1} \cdot C_s}{a \cdot C_{m'} + C_{m'} - C_s} = \frac{a}{a+1}$$

Consequently, $P'_f = P_f$. ■

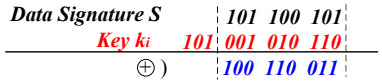
On the other hand, a concern for this jump pointer technique is the potential information leaking to the hackers, if they know p_j points to a data frame that shares some common attribute values. Note that the hacker does not have a way to obtain this jump pointer from signatures if the jump pointer is maintained as part of the encrypted information frame. Nevertheless, we analyze its potential impact under a worst case scenario, where the jump pointer is known to the hackers. The jump pointer connects two signatures S_i and S_j with $A_{S_i} \cap A_{S_j} \neq \emptyset$ via p_j. A hacker, in addition to performing the dictionary attack described in Section IV-B, may attempt to guess the common attributes indexed by both signatures. Let G_{S_i}/G_{S_j} represent the sets of signatures from the hacker's dictionary that match S_i/S_j. $G_{S_i} \cap G_{S_j}$ gives a superset of all the attributes that indexed by both S_i and S_j. The experiments to be presented in Section VI will demonstrate that the knowledge of $G_{S_i} \cap G_{S_j} \supseteq A_{S_i} \cap A_{S_j}$ does not cause any serious information leakage.

C. XOR Signature

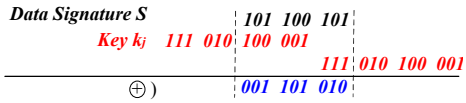
In the above subsections, we focus on the signature-based air index scheme in which only the data but not the index are encrypted. On the one hand, broadcasting index in a non-encrypted form simplifies the step of index retrieval and query processing. Without any expensive decryption operation, the consumption of power resource at client's side can be significantly reduced. On the other hand, it potentially leaves a door open to the attackers since the attackers can obtain some hints from the plain index. In this subsection, we present the XOR signature scheme to further reduce the information leakage without affecting the performance. In what follows, we present the basic idea in detail with an analytical model to derive the new false guess probability with confidentiality improvement.

The basic idea is motivated by the fact that a bit string, after two XOR (denoted as $\oplus$) operations with the same bit string, can be recovered[4]. In other words, $(A \oplus B) \oplus B = A \oplus (B \oplus B) = A \oplus 0 = A$. If there is a string B which is only known to authorized clients and servers, but not attackers, each signature S broadcast on the channel can be encrypted based on string B to form a new bit string $S' = S \oplus B$. When clients receive the index, the original signature S can be recovered by XOR with B, i.e., $S' \oplus B = S$. Note that the real data items in our model are encrypted, and each client knows the keys to decrypt the data. Therefore, the secret keys k_i are the best candidates for string B which do not incur any extra management overhead. Furthermore, the XOR operation between two strings A and B only needs to compare two strings bit by bit once that is much cheaper than the normal decryption process. Our simulations, based on a real PDA to be presented in Section VI-D, will further demonstrate the difference.

[4]XOR represents a operation on two operands that results in a value of *true* if and only if two operands share different values, i.e., $1 \oplus 1 = 0$, $0 \oplus 0 = 0$, $1 \oplus 0 = 1$, and $0 \oplus 1 = 1$.



(a) Example 1: $j = 3$



(b) Example 2: $j = 6$

Fig. 5. Encrypt signatures based on xor operation

Figure 5 shows two examples for signature encryption based on XOR operation, with the alignment position changing. Suppose there are a list of keys k_i used in the broadcast system, and a dynamic parameter j is set to $(t_i \cdot r)\%K$. Here, t_i means the time when the i-th frame in a broadcast cycle is going to be broadcast, r is a system-wide parameter, and K represents the number of keys. Consequently, we can reuse k_j to encrypt the signature. We further vary the alignment position dynamically to make the recovering of keys harder, if not impossible. For example, as shown in Figure 5(a), the j-th bit of the key k_j is aligning with the first bit of the signature, with $j = 3$, i.e., $101100101 \oplus 001010110 = 100110011$. Instead of broadcasting the plain signature 101100101, the server broadcasts the encrypted signature 100110011 and the authorized users can recover the original signature using the key k_3. In case that after alignment, the key k_j does not have enough remaining bits to match with the signature, we can expand k_j by appending the same key at the end of k_j. An example is depicted in Figure 5(b).

It is obvious that the confidentiality improvement does not change the setting of the signature. All the users can recover the original signature index based on XOR operation. As a result, it does not affect the false drop probability and hence the performance. On the other hand, the plain signature S is covered by the secret key k_j. Consequently, it is much harder for the attackers to guess the right information from the encrypted signature S' (= $S \oplus k_j$). If the attacker only knows the hash function to produce the signature, what he can do is to conduct a dictionary attack based on the received signatures S's, as described in Section IV-B. According to XOR operation, a bit set to 1 in the original signature S will remain the value 1 only when the corresponding bit of the secret key is set to 0. Without loss of generality, we assume the secret key has equal numbers of bits 1 and 0, i.e., only $m/2$ bits over the m-bit of the key are set to 0. Therefore, the possibility that w_b bits set to 1 corresponding to an indexed attribute can remain the value 1 even after the XOR operation is $(\frac{1}{2})^{w_b}$. As a result, the number of true guess G'_t can be approximated as follows.

$$G'_t = \begin{cases} (\frac{1}{2})^{w_b} \times s \times u & w_b \leq \frac{m}{2} \\ 0 & \text{otherwise} \end{cases}$$

Furthermore, XOR a bit with value 0 from the original signature S and a bit with value 1 from the secret key can also produce 1. On average, there are $(m - w_f)$ bits with value 0 in S, and the corresponding bits in the key on average have 50% probability to be 1. As a result, the total number of bits set to 1 in S'

is $\frac{w_f}{2} + \frac{m-w_f}{2} = \frac{m}{2}$. The total number of matched guess G' thereafter can be approximated as follows. It is worth noticing that when w_b exceeds the value of $\frac{m}{2}$, no match is found. This is because a received signature S', whose total number of bits set to 1 is guaranteed to be $\frac{m}{2}$, can not match a signature in hacker's dictionary with w_b larger than $\frac{m}{2}$.

$$G' = \begin{cases} \frac{\mathsf{comb}(\frac{w_f}{2}+\frac{m-w_f}{2}, w_b)}{\mathsf{comb}(m, w_b)} \times |D_H| & w_b \leq \frac{m}{2} \\ 0 & \text{otherwise} \end{cases}$$

Finally, the false guess probability P'_g with confidentiality improvement can be derived as follows.

$$P'_g = \frac{(G' - G'_t)}{|D_H| - G'_t} \tag{20}$$

In order to provide a complete description, we further assume that attackers are so smart that they successfully guess that the received signatures S' are the XOR results of the real signatures S and some other strings, i.e., $S' = S \oplus A$. Since the string A must be known to both the server and all the clients, attackers can further guess that different Ss are encrypted based on one string A in order to minimize the management cost of string A. Consequently, they may be able to remove string A by combining two received signatures S'_1 and S'_2 based on XOR operation, i.e., $S'_1 \oplus S'_2 = (S_1 \oplus A) \oplus (S_2 \oplus A) = (S_1 \oplus S_2) \oplus (A \oplus A) = (S_1 \oplus S_2) \oplus 0 = (S_1 \oplus S_2)$. Although our system adopts different keys and different alignments to encrypt the signatures, we assume a worst case scenario in which the XOR result of some signatures S_1 and S_2 can be recovered. However, even with this knowledge, the attacker still has to suffer from a higher false guess probability. This is because for a given bit string S with m bits, there are 2^m different strings that can produce S based on XOR operation. As a result, it is almost impossible for an attacker to guess the right answer, especially when m is sufficiently large.

VI. Experimental Evaluation

As discussed earlier, a signature-based wireless broadcast system can be flexibly tuned to meet different performance/confidentiality requirements. In this section, we conduct extensive experiments via simulations to verify the analytical results, and demonstrate the flexibility of signature-based air index. All the experiments are implemented in C language in a Unix system. As shown in Table II, the application domain D_A has $1,000$ attributes and we assume the broadcast cycle consists of $10,000$ data items, with each item characterized by u indexed attributes randomly drawn from D_A. On the other hand, the attacker's dictionary D_H contains $10,000$ attributes which is a superset of D_A (i.e., we made a conservative assumption from the administrator's standpoint). The signature size, m, is set to 64, 128, and 256 bits and the content body of each item, n, is set to 1280 bits, with $\frac{m}{n}$=5%/10%/20%. Although in the real applications the size of the signature and the content body might be much larger, we believe $\frac{m}{n}$=5%/10%/20% is consistent with many real applications. Consequently, our parameter settings simulate representative cases of real applications. By changing the value of s and tuning w_b from 1 to m, the system administrator can generate different configurations of signatures.

The experimental results presented here are the average performance of 200 queries, each of which is based on a random attribute value drawn from D_A. The false drop probability P_f and false guess probability P_g are obtained by both experiments and analysis for validation. The access time/tuning time is obtained by Equation (1)/Equation (2) and the ILD for each broadcast data item is experimentally obtained by counting the number of correct guesses and the number of total guesses. We have conducted five sets of experiments. The first set of experiments is to validate the analytical results of signature-based air index. As it has been presented in the preliminary report [23], it is skipped for space saving. The second set of experiments is to tune the signature in order to balance the performance and confidentiality, the third and fourth sets of experiments are to evaluate the effectiveness of the jump pointer technique and the XOR signature, respectively. Finally, the last set of experiments is to implement the signature-based approach in a real PDA to verify that encryption is expensive and to validate that the signature-based secure broadcast technique can significantly cut down the power consumption.

TABLE II
Simulation settings

Notation	Description	Settings
$\|D_A\|$	domain size of index attributes of a given application A	1000
$\|D_H\|$	domain size of the hacker's dictionary	10000
u	number of attributed index in one item	$4, 8, 10, 12$
N	number of the data items included in a broadcast cycle	10000
m	number of bits in a signature	$64, 128, 256$
n	the size of a data item content body in the unit of bits	1280
s	number of data items in an integrated data group	$1, 2, 4$
w_b	number of 1's in an attribute's signature	[1,m]
P_s	selectivity of a query	0.01

A. Balancing Performance and Security

As we mentioned in Section V, tuning of w_b is more flexible than that of s. In this set of experiments, we first validate the analytical model proposed to achieve a good balance between confidentiality and performance with a fixed s. Thereafter, we present the results under different s values.

First, the relationship between C_{nor} and w_b for various m value with $s = 1$ is shown in Table III. As the access time is not affected by w_b, we set the δ value to 1 and hence only the tuning time performance but not the access time contributes to C_{nor}. It is observed that the analytical results approach the simulation results, especially for the C_{nor} values. For the detailed w_b setting, the simulation result does not match perfectly with the analytical results. The main reason is that our analytical model presented in Section V-A actually derives the best P_f (i.e., P_g) value that minimizes the cost C_{nor} and then based on the value of P_f to derive w_b. However, as reported previously in [23], there is a gap between the analytical P_f (P_g) and real P_f (P_g), and that difference causes the mismatch of w_b settings. However, the analytical results still can provide useful hints. In those systems with dynamic system settings, it is always very expensive to get the optimized w_b via simulation especially when m is large. Consequently, the analytical model provides an alternative tool for performance analysis and system planning.

Secondly, we verify the analytical model under different system requirements. Here, we change both the setting of α and the setting of δ to obtain a thorough understanding of the impact of both s and w_b. The optimized settings, together with C_{nor} values,

TABLE III

OPTIMAL CONFIGURATIONS OF SIGNATURE SCHEME ($s = 1$, $\delta = 1$)

m	α	Analysis Results		Simulation Results	
		w_b	C_{nor}	w_b	C_{nor}
64	0.25	6	−0.2101	5	−0.2103
	0.5	4	−0.4433	4	−0.4414
	0.75	4	−0.6851	4	−0.6728
128	0.25	31	−0.1976	26	−0.1956
	0.5	27	−0.4183	23	−0.4145
	0.75	22	−0.6475	21	−0.6411
256	0.25	88	−0.1726	72	−0.1788
	0.5	81	−0.3684	67	−0.3792
	0.75	76	−0.5727	63	−0.5873

of the system with $m = 64$ and $s \in \{1, 2, 4\}$ are presented in Table IV. We report the performance for each s value, and those in bold represent the settings that minimize the system cost. Initially, we set both α and δ to be pretty small, and hence the system cost is dominated by the confidentiality loss. Consequently, a system with small ILD (i.e., large false guess probability P_g) is preferred. That's why when we compare the optimal setting under $\alpha = \delta = 0.1$ with that under $\alpha = \delta = 0.5$, we observe the w_b value related to $\alpha = \delta = 0.1$ is much larger. Overall, the signature with $s = 1$ is preferred.

However, as α value increases, performance gains start to dominate the system cost. For example, when $\alpha = 0.9$ and $\delta = 0.1$, the signature with $s = 2$ provides the minimal cost. As we further increase the weight of performance, especially the importance of access time, the signature with $s = 4$ outperforms the best and becomes the optimal choice. This is consistent with our previous analysis. The value of s mainly affects the access time, while w_b affects the false guess probability/false drop probability and hence it determines the tuning time performance and ILD. When the short access time is the ONLY system requirement, a large s is preferred. On the other hand, when the confidentiality loss is the major system requirement, a small s is more suitable.

TABLE IV

OPTIMAL CONFIGURATIONS OF SIGNATURE SCHEMES ($s \in \{1, 2, 4\}$, AND $m = 64$)

α	δ	s	Analysis Results		Simulation Results	
			w_b	C_{nor}	w_b	C_{nor}
0.10	0.10	4	3	0.0033	3	0.0034
		2	7	0.0011	7	0.0011
		1	**17**	0.0010	**15**	0.0010
0.50	0.50	4	1	−0.1219	1	−0.1174
		2	2	−0.1780	2	−0.1757
		1	**5**	−0.2026	**4**	−0.2029
0.90	0.10	4	1	−0.0354	1	−0.0339
		2	**2**	−0.0468	**2**	−0.0459
		1	4	−0.0387	4	−0.0387
0.99	0.01	**4**	**1**	0.0074	**1**	0.0075
		2	2	0.0172	2	0.0172
		1	4	0.0402	4	0.0402

B. Evaluation on Jump Pointer Technique

This set of experiments is to evaluate the performance of the jump pointer technique. The notation With Jump Pointer is to represent the signature-based air index with jump pointers, while Without Jump Pointer denotes the conventional signature-based air index. As we explain in Section V-B, a client has to scan all the C/s signatures to evaluate a query under the conventional signature-based approach. On the other hand, the jump pointer links each signature s to the next closest signature s' that shares common attributes with s. Consequently, once a query finds a true match with a signature s_i of the i-th data frame, it can follow the jump pointer to access the next signature s_j that might satisfy the query. As a result, the retrievals of signatures broadcast between the i-th data frame and the j-th data frame can be safely skipped.

Figure 6 depicts the average number of signature retrievals incurred by processing a query under different u setting, with m fixed at 64 and s fixed at 1. It is noticed that without any jump pointer, the number of signature retrievals equals $C/s = 10000$, and jump pointers can reduce the number of signatures checked. It is observed that the improvement is dependent on u, which is consistent with the analysis conducted in Section V-B. When u is small, it is less likely that two signatures share some common attributes and hence the average distance between a signature s and the one pointed by s's jump pointer is larger which results in a more significant saving of the signature retrievals.

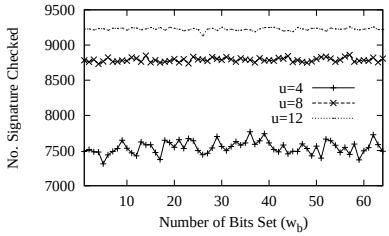
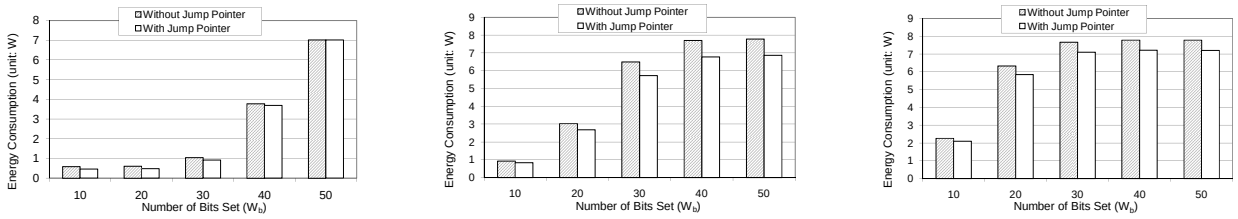


Fig. 6. No. of signatures retrieved with jump pointers ($m = 64$,$s = 1$)

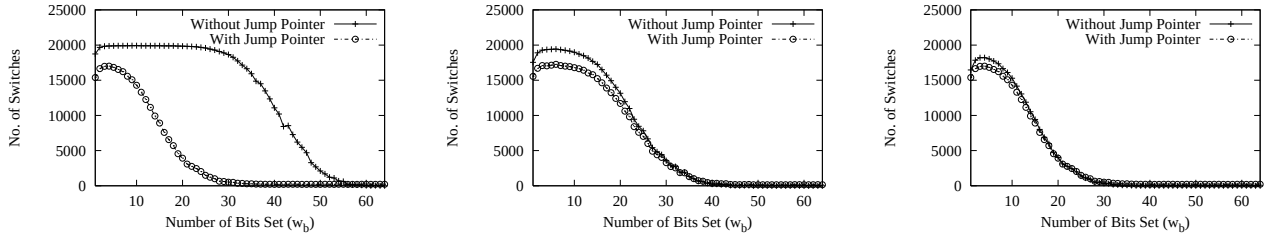
To more precisely demonstrate the advantage of the jump pointer technique in terms of energy conserving, we obtain the average power consumptions with/without jump pointers. We adopt the numbers from Proxim RangeLAN2 [24], which requires $1.5W$ in transmit mode, $0.75W$ in receive mode and $0.01W$ in doze mode, to approximate the energy consumption in retrieving signatures and data items. The average power consumption per query processing can be approximated by $\frac{TUNE}{B} \times 0.75 + \frac{(ACC-TUNE)}{B} \times 0.01$. Here, B represents the bandwidth of the broadcast channel whose value is set to 1Mbps. Figure 7 depicts the average energy consumption under different u/w_b values, with $m = 64$ and $s = 1$. As expected, the jump pointer technique can significantly reduce the energy consumption. For example, when $u = 4$ and $w_b = 10$, jump pointers can save energy consumption up to 23.5%, compared with that without jump pointers.

In the above analysis, we do not consider the switching cost, as most of the existing work. However, it is observed that the typical setup time for a mobile device to start or tune into active mode is in the order of $100\mu s$ [24], which might consume non-neglectable energy. As the jump pointer technique effectively reduces the number of signature retrievals, it can reduce the number of switches as well. Here, both a transition from the active mode to doze mode and vice versa are counted as a switch. Figure 8 plots the average switches incurred during the processing of one query, with $m = 64$ and $s = 1$, and Figure 9 depicts the average power consumption with switching cost considered. Here, we vary the setup time from $1\mu s$, to $10\mu s$, to $100\mu s$, and finally to $1ms$, and we assume it requires a mobile device $0.75W$ for setup, the same as in the receive mode. Compared with the energy consumption shown in Figure 7, the switching cost is obviously



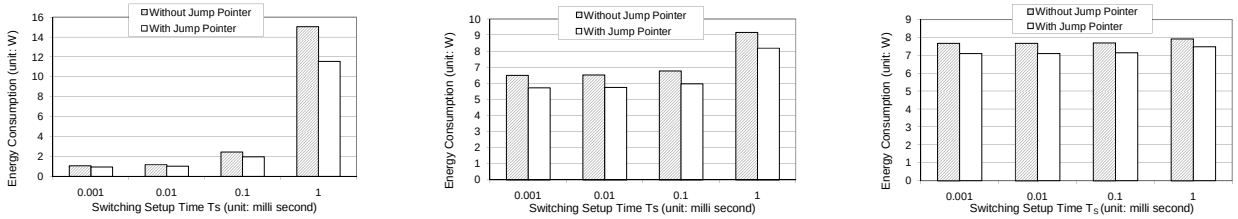
(a) $u = 4$ (b) $u = 8$ (c) $u = 12$

Fig. 7. Energy consumption with/without jump pointers ($m = 64$,$s = 1$)



(a) $u = 4$ (b) $u = 8$ (c) $u = 12$

Fig. 8. Average number of switches with/without jump pointers ($m = 64$,$s = 1$)



(a) $u = 4$ (b) $u = 8$ (c) $u = 12$

Fig. 9. Energy consumption with switching overhead ($m = 64$, $w_b = 30$, $s = 1$)

non-neglectable.

C. Evaluation on XOR Signature

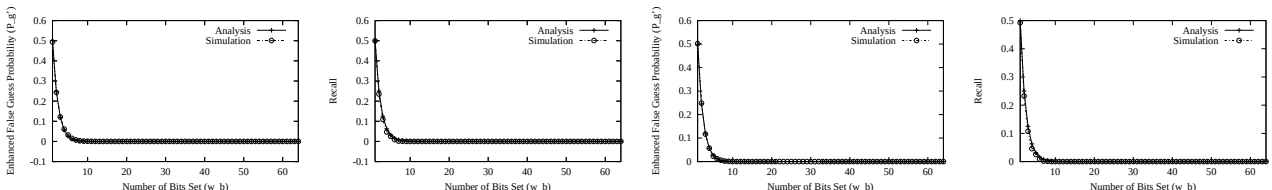
This set of experiments is conducted to validate the analytical results of the XOR signature approach, with all the system parameters sharing the same settings described in Table II. It is observed that pre-defined metric ILD is no longer suitable with confidentiality improvement, because in many cases the number of matched guesses G is zero. Consequently, a new metric, namely *recall*, is defined as the ratio of the number of right guesses to the total number of attributes indexed by the signature. In other words, recall equals the percentage of the indexed attributes that are known to the hackers, and a small recall indicates a high confidentiality.

Figure 10 shows the experimental results, as well as the analytical results when m equals 64. It is observed that the approximated results obtained from theoretical analysis match the simulation results perfectly. Take Figure 10(a) as an example, the average difference between analytical and experimental results is around 4×10^{-6}. The second observation is that the false guess probability P'_g in most cases is zero. In Section V-A, we have claimed that an ideal broadcast system prefers a small tuning time/access time with large false guess probability. Does this mean that the XOR signature with an almost-zero false guess probability leaks more useful information to the attackers? The answer is definitely *NO*. Although the almost-zero false guess probability reflects a fact that the number of false guesses G'_f is zero, it ignores the fact that the total number of matched guesses G' is almost zero as well. That's why we introduce the new metric $recall$ to capture the number of true guesses G'_t, which can help to derive the total number of guesses $G' = G'_t + G'_f = G'_t + P'_g \times (|D_H| - G'_t) = recall \times s \times u + P'_g \times (|D_H| - recall \times s \times u)$.

The third observation is that $recall$ is almost zero in the XOR signature scheme, as shown in Figure 10(b) and Figure 10(d). In normal signature scheme, we assume all the indexed attributes are available in hacker's dictionary. In other words, the number of true guesses always equals the number of attributes indexed by a signature, i.e., $G_t = s \times u$. Therefore, the $recall$ is always 100%. However, it can successfully confuse the hacker by the XOR operation and the number of true guesses, in most cases, is zero. Both theoretical analysis and experiments show that the XOR signature scheme effectively prevents the attackers from getting useful information from the received signature. Under open (i.e., non-encrypted) signature scheme, the system does not try to avoid the true match from the hackers. Instead, it tries to tune the inherited false drop from the signature scheme to lower the percentage of the true matches over the total matches. On the other hand, the XOR signature scheme prevents the true match from happening at the first place. In our evaluation, the performance results under $m = 128$ and $m = 256$ are similar and hence are not presented to save space.

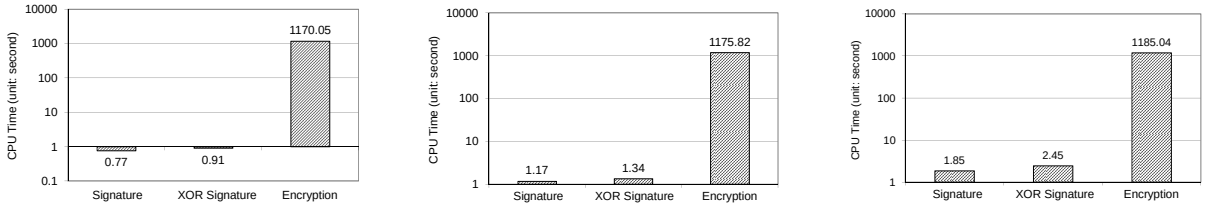
D. Empirical Evaluation on PDA

The focus of this paper is to employ and tune the signature-based air index to balance confidentiality and performance in



(a) Enhanced P_g' ($s = 1$) (b) Recall ($s = 1$) (c) Enhanced P_g' ($s = 4$) (d) Recall ($s = 4$)

Fig. 10. Security of the XOR signature scheme ($m = 64$,$u = 10$)



(a) $m = 64$ (b) $m = 128$ (c) $m = 256$

Fig. 11. CPU time for running 10^6 queries of different approaches ($s = 1$)

wireless broadcast systems. An arising question is why not broadcast encrypted index information directly which will eliminate most of, if not all, confidentiality concerns. The main reason is that the decryption performed at the thin clients (e.g., PDA or mobile phone) is very expensive. While encryption/decryption can address some security concerns, it deteriorates performance significantly. In order to demonstrate the inefficiency of encryption/decryption operation and the significant improvement brought by signature-based schemes, we implement the algorithms in a real mobile device, SHARP PDA (as shown in Figure 12(a)). The detailed system specification of the PDA is listed in Figure 12(b). OpenSSL toolkit (http://www.openssl.org/) is used to implement encryption/decription operations.

Three schemes, namely *encryption*, *signature*, and *XOR signature*, are evaluated. In all three algorithms, the server interleaves index and data frames on the broadcast channel. The client is supposed to check index first before spending time retrieving the real data, which is much larger than index in size. We further assume all the registered clients share a key with the server, and the server encrypts all the data using that key. The first algorithm encrypts index, as well as all the data, using this common key. When a client issues a query, it downloads the index, decrypts it, and compares it against its query. The other two schemes only encrypt data, but not index. Signature scheme broadcasts plain index (i.e., signature of the corresponding data), and XOR signature scheme broadcasts the XOR product of the plain signature and a key. Once the index is downloaded, the client can compare the query with the plain index directly without any additional operation in signature scheme. On the other hand, clients first need to XOR the key with retrieved index to remove the shield from the key, and then compare the result with their queries under XOR signature scheme. In all three schemes, the retrieval of the data is triggered only when a match between the query and signature is detected.

Collecting power consumption data is not easy. As a result, in this set of experiments, we evaluate the CPU time used by different approaches instead of measuring the power consumption directly. The total CPU cost of running 1 million (10^6) queries of different approaches is shown in Figure 11. It is obvious that the encryption/decryption approach incurs a much higher CPU

(a)

OS	Linux
Processor	Intel XScale PXA270
Processor Speed	416MHz
Hard Drive	4GB
Installed ROM	16MB
Installed RAM	64MB
Battery Type	Lithium ion

(b) Product Specification

Fig. 12. SHARP Zaurus SL_C3100

cost, around 1054.03 times larger than that of signature approach and 881.31 times larger than that of XOR signature approach. In addition, we can find out that the larger the m is, the longer the CPU time is, which is consistent with our expectation. The value of m plays a more important role on the CPU time of the signature approach and the XOR signature approach, compared with the encryption/decryption approach. The reason behind is that the decryption algorithms in general are very complicated and reducing m value does not simplify the operation. On the other hand, the signature scheme and the XOR signature scheme only employ very simple bit-operations (e.g., AND/XOR). This observation well justifies our argument that encryption/decryption significantly affects the performance. On the other hand, both the signature approach and the XOR signature approach employ very simple bit operations and therefore the incurred computation cost is very low, which guarantees the efficiency of the approaches.

VII. Conclusion

Air indexing is an important technique that facilitates energy conservation of mobile clients in wireless broadcast systems. However, the crucial confidentiality issues on air indexing have not been discussed in the literature. This study, to the best of our knowledge, is the first research effort to address both performance and confidentiality issues via indexing techniques in wireless data broadcast systems.

In this paper, we argue that signature-based air index is an ideal technique to meet the performance and confidentiality requirements of applications because the tradeoff between performance and confidentiality metrics can be properly tuned by system administrators. We define a security metrics called information leaking degree to measure confidentiality loss in air indexes and analyze both confidentiality and performance metrics in terms of a number of controllable parameters. We further propose the jump pointer technique and the XOR signature scheme to improve the performance and the confidentiality, respectively.

This is a new research direction which deserves more effort from the research community. We are developing new air indexing techniques for secure wireless data broadcast and performing more detailed analysis. Both of the performance and confidentiality aspects of wireless data broadcast will be further exploited in our future study.

Acknowledgment

The authors would like to thank Qingzhao Tan for her contribution to this project in the early stage. In this research, Dik Lun Lee was supported in part by Research Grant Council, Hong Kong under Grant no. 615707 and CA05/06.EG03. Wang-Chien Lee was supported in part by the National Science Foundation under Grant no. IIS-0328881, IIS-0534343 and CNS-0626709.

References

[1] S. Acharya, M. Franklin, and S. Zdonik. Balancing push and pull for data broadcast. In *Proc. ACM SIGMOD International Conference on Management of Data (SIGMOD'97)*, pp. 183-194, May 1997.

[2] A. R. Beresford and F. Stajano. Location privacy in pervasive computing. *IEEE Pervasive Computing*, 1(3), 2003.

[3] B. Bloom. Space/time trade-offs in hash coding with allowable errors. *Communications of the ACM (CACM)*, 13(7), July 1970.

[4] D. Djenouri, L. Khelladi, and A. N. Badache. A survey of security issues in mobile ad hoc and sensor networks. *IEEE Communications Surveys and Tutorials*, 7(4):2-28, 2005.

[5] W. Du, J. Deng, Y. Han, and P. Varshney. A pairwise key pre-distribution scheme for wireless sensor networks. In *Proc. 10th ACM conference on Computer and communications security (CCS'03)*, pp. 42-51, 2003.

[6] L. Eschenauer and V. Gligor. A key-management scheme for distributed sensor networks. In *Proc. 9th ACM conference on Computer and communications security (CCS'02)*, pp. 41-47, 2002.

[7] B. Gedik and L. Liu. Location privacy in mobile systems: A personalized anonymization model. In *Proc. 25th IEEE International Conference on Distributed Computing Systems (ICDCS'05)*, pp. 620-629, June 2005.

[8] M. Gruteser and D. Grunwald. Anonymous usage of location-based services through spatial and temporal cloaking. In *Proc. 1st International Conference on Mobile Systems, Applications, and Services (MobiSys'03)*, May 2003.

[9] Q. Hu, W.-C. Lee, and D. Lee. Indexing techniques for wireless data broadcast under clustering and scheduling. In *The Eighth International Conference on Information and Knowledge Management (CIKM'99)*, November 2-6 1999.

[10] Q. Hu, W.-C. Lee, and D. Lee. A hybrid index technique for power efficient data broadcast. *Distributed and Parallel Databases (DPDB)*, 9(2):151-177, March 2001.

[11] Y. Huang and W. Lee. A cooperative intrusion detection system for ad hoc networks. In *Proc. 1st ACM workshop on Security of ad hoc and sensor networks (SASN'03)*, pp. 135-147, 2003.

[12] T. Imielinski, S. Viswanathan, and B. R. Badrinath. Energy efficiency indexing on air. In *Proc. ACM SIGMOD International Conference on Management of Data (SIGMOD'94)*, pp. 25-36, May 1994.

[13] T. Imielinski, S. Viswanathan, and B. R. Badrinath. Power efficiency filtering of data on air. In *Proc. 4th International Conference on Extending Database Technology (EDBT'94)*, pp. 245-258, March 1994.

[14] T. Imielinski, S. Viswanathan, and B. R. Badrinath. Data on air - organization and access. *IEEE Trans. Knowledge and Data Engineering (TKDE)*, 9(3):353-372, May-June 1997.

[15] D. Lee and C. Leng. Partitioned signature file: Design considerations and performance evaluation. *ACM Trans. Information Systems (TOIS)*, 7(2):158-180, 1989.

[16] K. C. K. Lee, H. V. Leong, and A. Si. A semantic broadcast scheme for a mobile environment based on dynamic chunking. In *Proc. 20th IEEE International Conference on Distributed Computing Systems (ICDCS'00)*, pp. 522-529, April 2000.

[17] W.-C. Lee and D. Lee. Using signature and caching techniques for information filtering in wireless and mobile environments. *Journal of Distributed and Parallel Databases*, 4(3):57-67, 1996.

[18] C. Leng and D. Lee. Optimal weight assignment for signature generation. *ACM Trans. Database Systems (TODS)*, 17(2):346-373, 1992.

[19] D. Liu and P. Ning. Tinysec: a link layer security architecture for wireless sensor networks. In *Proc. 2nd international conference on Embedded networked sensor systems (SenSys'04)*, pp. 162-175, 2004.

[20] S. Marti, T. J. Giuli, K. Lai, and M. Baker. Mitigating routing misbehavior in mobile ad hoc networks. In *Proc. 6th International Conference on Mobile Computing and Networking (MobiCom'00)*, pp. 255-265, 2000.

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

[22] K. Tan and J. X. Yu. Generating broadcast programs that support range queries. *IEEE Trans. Knowledge and Data Engineering (TKDE)*, 10(4):668-672, 1998.

[23] Q. Tan, W.-C. Lee, B. Zheng, P. Liu, and D. Lee. Balancing performance and confidentiality in air index. In *Proc. ACM 14th Conference on Information and Knowledge Management (CIKM'05)*, pp. 800-807, October 2005.

[24] A. Wang, S. Cho, G. Sodini, and P. Chandrakasan. Energy efficient modulation and mac for asymmetric rf microsensor systems. In *Proc. international symp. Low Power Electronics and Design (ISLPED'01)*, pp. 106-111, 2001.

[25] J. Xu, B. Zheng W-C Lee, and D. Lee. The d-tree: An index structure for planar point queries in location-based wireless services. *IEEE Trans. Knowledge and Data Engineering (TKDE)*, 16(12):1526-1542, 2004.

[26] T. W. Yan and H. García-Molina. Index structures for selective dissemination of information under the boolean model. *ACM Trans. Database Systems (TODS)*, 19(2):332-364, 1994.

[27] B. Zheng and D. Lee. Information dessimination via wireless broadcast. *Communications of the ACM*, 48(5):105-110, May 2005.

[28] B. Zheng, J. Xu, W. C. Lee, and D. L. Lee. Grid-partition index: A hybrid approach to nearest-neighbor queries in wireless location-based services. *VLDB Journal,*, 15(1):21-39, 2006.