ISSN : 2229-4333(Print) | ISSN : 0976-8491(Online)

Securing Supply Chain Management System using RFID

[1]P. Mohana Priya , [2]B. Sangeetha, [3]V.G. Vijaya Kumaran
[1,2,3]Karpagam University, Coimbtore, Tamilnadu, India

Abstract
RFID Applications can be used to monitor and manage the movement of the finished products throughout a supply chain. RFID tags can be attached directly to the items and materials or they can be attached to the containers that carry them. Pallets, trailers, totes, carts, cargo containers, and reusable transport items can all be tagged. Readers placed throughout a facility can monitor movement and location of inventory, thus providing real time data. Since RFID tags can be used to identify each individual item, enormous amounts of location-tracking data are generated. With such data, object movements can be modeled by movement graphs, where nodes correspond to locations and edges record the history of item transitions between locations. Especially this study has been taken to develop a movement graph model as a compact representation of RFID data sets. This can be within a warehouse, a freight yard or within a retail location. RFID applications in the supply chain enable more frequent and accurate inventory counts RFID applications in the supply chain can also decrease costs associated with inventory counting. This study show that such a graph can be better organized around gateway nodes, which serve as bridges connecting different regions of the movement graph.

Keywords
RFID, Security, location tracking, monitor.

I. Introduction

The increasingly wide adoption of RFID technology by retailers to track containers, pallets, and even individual items as they move through the global supply chain, from factories in exporting countries, through transportation ports, and finally to stores in importing countries, creates enormous data sets containing rich multidimensional information on the movement patterns associated with objects and their characteristics. However, this information is usually hidden in terabytes of low-level RFID readings, making it difficult for data analysts to gain insight into the set of interesting patterns influencing the operation and efficiency of the procurement process. In order to realize the full benefits of detailed object tracking information, we need to develop a compact and efficient RFID cube model that provides OLAP-style operators useful to navigate through the movement data at different levels of abstraction of both spatiotemporal and item information dimensions. This is a challenging problem that cannot be efficiently solved by traditional data cube operators, as RFID data sets require the aggregation of high-dimensional graphs representing object movements, not just that of entries in a flat fact table.

We propose to model the RFID data warehouse using a movement graph-centric view, which makes the warehouse conceptually clear, better organized, and obtaining significantly deeper compression and performance gain over competing models in the processing of path queries. The importance of the movement graph approach to RFID data warehousing can be illustrated with an example.

Example. Consider a large retailer with a global supplier and distribution network that spans several countries and that tracks objects with RFID tags placed at the item level. Such a retailer sells millions of items per day through thousands of stores around the world, and for each such item, it records the complete set of movements between locations, starting at factories in producing countries, going through the transportation network, and finally arriving at a particular store where the item is purchased by a customer. The complete path traversed by each item can be quite long as readers are placed at very specific locations within factories, ships, and stores (e.g., a production lane, a particular truck, or an individual shelf inside a store). Further, for each object movement, proper-ties such as shipping cost, temperature, and humidity can be recorded.

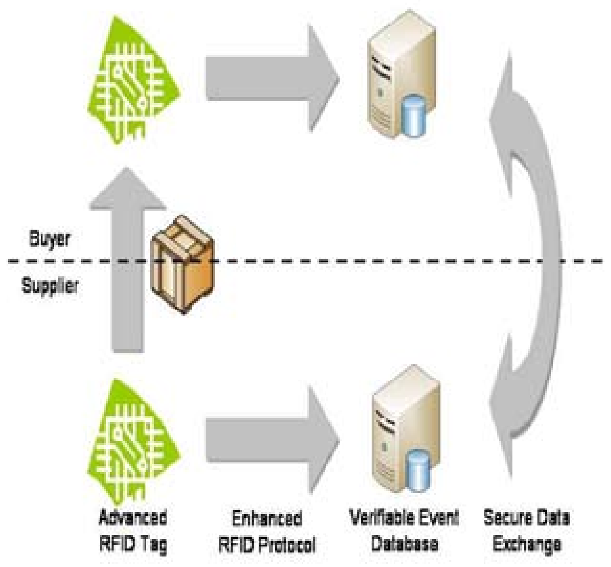


Fig. 1: Basic Architecture of Track and Trace

The questions become "how can we present a clean and well-organized picture about RFID objects and their movements?" and "whether such a picture may facilitate data compression, data cleaning, query processing, multilevel, multidimensional OLAPing, and data mining?"

Our movement graph approach provides a nice and clean picture for modeling RFID objects at multiple levels of abstraction. And it facilitates data compression, data cleaning, and answering rather sophisticated queries, such as . Top-level combined/OLAP query: What is the average shipping cost of transporting electronic goods from factories in Hyderabad to stores in Japan in 2007? Path query: Print the transportation paths for fish products from Cochin sold in Kashmir on 18 September that were exposed to over 40 degree centigrade heat for over 5 hours on the route.

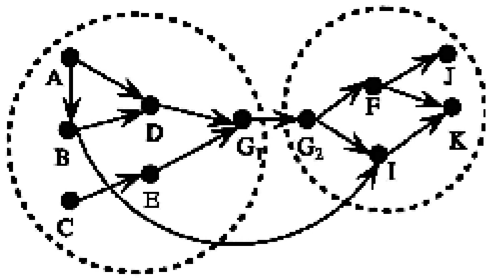


Fig. 2: An example movement graph.

We propose a movement graph-based model, which leads to concise and clean modeling of massive RFID data sets and facilitates RFID data compression, query answering, cubing, and data mining. The movement graph is a graph that contains a node for every distinct (or more exactly, interesting) location in the system, and edges between locations record the history of shipments (groups of items that travel together) between locations. For each shipment we record a set of interesting measures such as travel time, transportation cost, or sensor readings such as temperature or humidity. We show that this graph can be partitioned and materialized according to its topology to speed up a large number of queries, and that it can be aggregated into cuboids at different abstraction levels according to location, time, and item dimensions, to provide multidimensional and multilevel summaries of item movements.

Fig. 1 shows Goods equipped with RFID tags pass from the supplier to the buyer. Each company reads the RFID tags and stores related information, an event, in its local database. Later the companies exchange that information in order to run advanced applications.

Fig. 2 summarizes our proposed data warehousing architecture. We receive as input a sequence of RFID readings and store them in database, recording the path traversed by each item. Based on the structure of paths, we construct a movement graph, which is partitioned along gateway nodes. Each partition is then cubed independently, and a query processing module answers OLAP queries over the aggregated movement graph.

The technical assistance can be summarized as follows:

A. Gateway based partitioning of the movement graph

We make the key observation that the movement graph can be divided into disjoint partitions that are connected through special gateway nodes. Most paths with locations in more than one partition include the gateway nodes. For example, most items travel from India to the China by going through major shipping ports in both countries. Gateways can be given by a user or be discovered by analyzing traffic patterns. Further, materialization can be performed for indirect edges between individual locations and their corresponding gateways and between gateways. Such materialization facilitates computing measures on the paths connecting locations in different partitions. An efficient graph partitioning algorithm, is developed, that uses the gateways to split the graph into clusters in a single scan of the path database.

B. Redundancy elimination compression

RFID readers provide tuples of the form (EPC, location, time) at fixed time intervals. When an item stays at the same location, for a period of time, multiple tuples will be generated. We can group these tuples into a single one of the form (EPC, location, time in, time out). This form of compression is lossless, and it can significantly reduce the size of the raw RFID readings.

C. Partition-based bulky movement simplification

Items tend to move and stay together through different locations. For example, a pallet with 500 cases of pen drives may arrive at the warehouse; from there, cases of 50 pen drives may move to the shelf; and from there, packs of 5 pen drives may move to the checkout counter. We can register a single stay or transition record for thousands of items that stay and move together. Such a record would point to a gid, which is a generalized identifier pointing to the subgroups that it contains. In global supply chain applications, one may observe a “merge-split” process, e.g., shipments grow in size as they approach the major ports, and then after a long distance bulky shipping, they gradually split (or even recombine) when approaching stores. We propose a partitioned map table that creates a separate mapping for each partition, rooted at major shipping ports.

D. Movement graph aggregation

The movement graph can be aggregated to different levels of abstraction according to the location concept hierarchy that determines which subset of locations are interesting for analysis, and at which level. For example, the movement graph may only have locations inside Massachusetts, and it may aggregate every individual location inside factories, ware-houses, and stores to a single node. This aggregation mechanism is very different from the one present in traditional data cubes as nodes and edges in the graph can be merged or collapsed, and the shipments along edges and their measures need to be recomputed using different semantics for the cases of node merging and node collapsing.

II RFID DATA

In this section, we give a brief introduction to data generation in RFID applications and explain the difficulties of applying traditional data warehousing models to such data.

A. Data Generation

An RFID object tracking system is composed of a collection of RFID readers scanning for tags at periodic intervals. Each such reader is associated with a location, and generates a stream of time-ordered tuples of the form (EPC, location, time), where EPC [1] is a unique 96-bit electronic product code associated with a particular item, location is the place where item was detected, and time is the time when the detection took place. Significant data compression is possible by merging all the readings for an item that stays at a location for a period of time, into a tuple of the form (EPC, location, time in, time out), where time in is the time when the item identified by EPC entered location and time out is the time when it left.

By sorting tag readings on EPC we can generate a path database, where we store the sequence of locations traversed by each item. Entries in the path database are of the form:
(EPC,(l1, time in1,time out1))l2,time in2,time out2) . . . (lk,time ink,time outk)).

In addition to the location information collected by RFID readers, an RFID application has detailed information on the characteristics of each item in the system, such information can be represented with tuples of the form (PC, d_1, d_2; . . . ; d_m) where each di is a particular value for dimension D_i, and typical dimensions could be product type, manufacturer, weight, or price. In many cases we can also have extra information describing measurements or proper-ties collected during item shipments, this data has the form (from; t_o; t_1; t_2; tag list: measure list) where from and to are the initial and final locations of the shipment, t_1 and t_2 are the starting and ending time of the shipment, tag list is the set of items transported, and measure list describes properties such as temperature, humidity, or shipping cost.

B. Data Cubing Challenges

The path nature of RFID data makes it hard to incorporate into a traditional data cube while preserving its structure. Suppose we view the cleansed RFID data as the fact table with dimensions (EPC; location; time in; time out: measure). The data cube will compute all possible group-bys on this fact table by aggregating records that share the same values (or any value, represented by the symbol *) at all possible combinations of dimensions. If we use count as measure, we can get, for example, the number of items that stayed at a given location for a given month. The problem with this form of aggregation is that it does not consider links between the records. For example, if we want to get the number of items of type "dairy product" that traveled from the distribution center in Delhi to stores in German, we cannot get this information. We have the count of "dairy products" for each location but we do not know how many of these items went from the first location to the second. The problem could be solved by increasing the number of dimensions, we could use a separate dimensions for distinct path segments; cells would then contain multi location aggregates. The problem with this approach is that as we increase the number of dimensions, the size of the data cube grows exponentially. We need a model capable of aggregating RFID data concisely while preserving its path-like structure for OLAP analysis.

III. Data Compression

A. Redundancy Removal Compression

RFID data contains large amounts of redundancy. Each reader scans for items at periodic intervals, and thus, generates hundreds or even thousands of duplicate readings for items in its range, which are not moving. For example, if a pallet stays at a warehouse for 7 days, and the reader scans for items every 30 seconds, there will be 20,160 readings of the form (EPC, warehouse, time). We could compress all these readings, without loss of information, to a single tuple of the form (EPC; warehouse, time in, time out), where time in is the first time that the EPC was detected in the warehouse and time out the last one.

Redundancy elimination can be accomplished by sorting the raw data on EPC and time, and generating time in and time out for each location by merging consecutive records for the same object staying at the same location.

B. Massive Movement Compression

Since a large number of items travel and stay together through several stages, it is important to represent such a collective movement by a single record no matter how many items were originally collected. As an example, if 1,000 boxes of biscuits stayed in location locA between time t1 (time in) and t2 (time out), it would be advantageous if only one record is registered in the database rather than 1,000 individual RFID records. The record would have the form (gid; prod; locA; t_1; t_2; 1;000), where 1,000 is the count, prod is the product id, and gid is a generalized id which will not point to the 1,000 original EPCs but instead point to the set of new gids which the current set of objects move to. For example, if this current set of objects were split into 10 partitions, each moving to one distinct location, gid will point to 10 distinct new gids, each representing a record. The process iterates until the end of the object movement where the concrete EPCs will be registered. By doing so, no information is lost but the number of records to store such information is significantly reduced.

The process of selecting the most efficient grouping for items, both in terms of compression and query processing, depends on the movement graph topology.

C. Split-Only Model

In some applications, the movement graph presents a tree-like structure, with a few factories near the root, ware-houses and distribution centers in the middle, and a large number of individual stores at the leaves. In such topology, it is common to observe items moving in large groups near the factories and splitting into smaller groups as they approach individual stores. We say that movement graphs with this topology present a Split-Only model of object movements.

In the Split-Only model, we can gain significant compression by creating a hierarchy of gids, rooted at factories where items move in the largest possible groups, and pointing to successively smaller groups as items move down the supply chain. In this model, a single grouping schema provides good compression because the basic groups, in which objects move, are preserved throughout the different locations, i.e., the smallest groups that reach the stores are never shuffled, but are preserved all the way from the Factory. In the next section, we will present a more general model that can accommodate both split and merging of groups.

D. Merge-Split Model

A more complex model of object movements is observed in a global supply chain operation, where items may merge, split, and groups of items can be shuffled several times. One such case is when items move between exporting and importing countries. At the exporting country, items merge into successively large groups in their way from factories to logistic centers, and finally to large shipping ports. In the importing country, the process is usually reversed, items split into successively smaller groups as they move from the incoming port, to distribution centers, and all the way to individual stores. We say that movement graphs with this topology present a Merge-Split model of object movements.

A single object grouping model, such as the one used in a Split-Only model would not be optimal when groups of items can both split and merge. A better option is to partition the movement graph around gateways, and define an item grouping model at the partition level. For example, the exporting country would get a hierarchy of groups rooted at the port and ending at the factories, while the importing country will have a separate hierarchy rooted at the port and ending at the individual stores. Using a single grouping for both partitions has the problem that each group would have to point to many small subgroups, or even just individual items that are preserved throughout the entire supply chain, after multiple operations of merge, split, and shuffle. Separate groupings prevent this problem by requiring bulky movement only at the partition level, and allowing for merge, split, and even shuffling of items without loss of compression.

E. Data Generalization

Since many users are only interested in data at a relatively high abstraction level, data simplification can be explored to group, merge, and compress data records. This type of simplification as opposed to the previous two simplification methods is lossy, because once we aggregate the data at a high level of abstraction, e.g., time aggregated from second to hour; we cannot ask queries for any level below the aggregated one.

There are two types of data generalization: item-based, which the same is encountered in traditional data cubes and does not involve spatiotemporal dimensions; and path-based, which is unique to RFID data sets.

Path Level Generalization. A new type of data generalization, not present in traditional data cubes, is that of merging and collapsing path stages according to time and location concept hierarchies. For example, if the minimal granularity of time is hour, then objects moving within the same hour can be seen as moving together and be merged into one movement. Similarly, if the granularity of the location is shelf, objects moving to the different layers of a shelf can be seen as moving to the same shelf and be merged into one.

Another type of path generalization is that of expanding different types of locations to different levels of abstraction, depending on the analysis task. For example, a transportation manager may want to collapse all movements inside stores and warehouses while expanding movements within trucks and transportation centers to a very detailed level. On the other hand, store managers may want to collapse all object movements outside their particular stores.
An important difference between path level generalization and the more conventional data cube generalization along concept hierarchies is that in path level aggregation, we need to preserve the path structure of the data, i.e., we need to make sure that the new times, locations, and transitions are consistent with the original data.

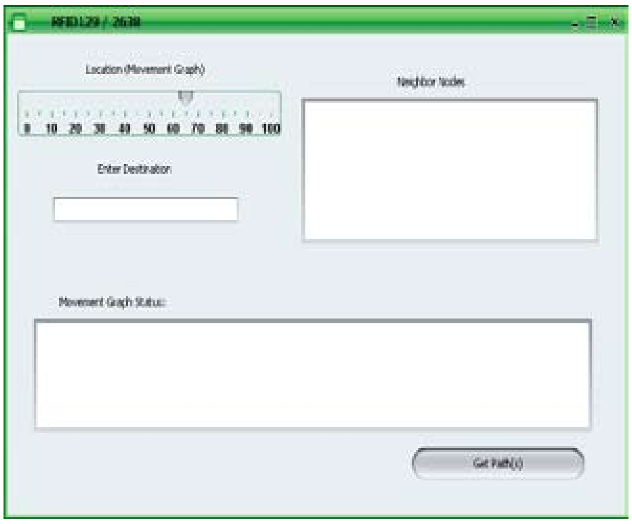
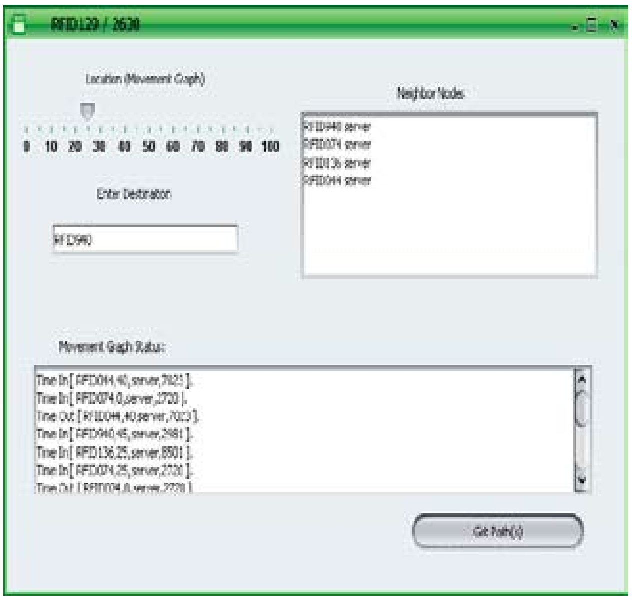


Fig. 3: Finding Location and Neighbor Nodes

Fig. 3 shows finding the goods location and the neighbor nodes of that corresponding location. It was calculated by using t_1 (time in) and t_2 (time out).



RFID940

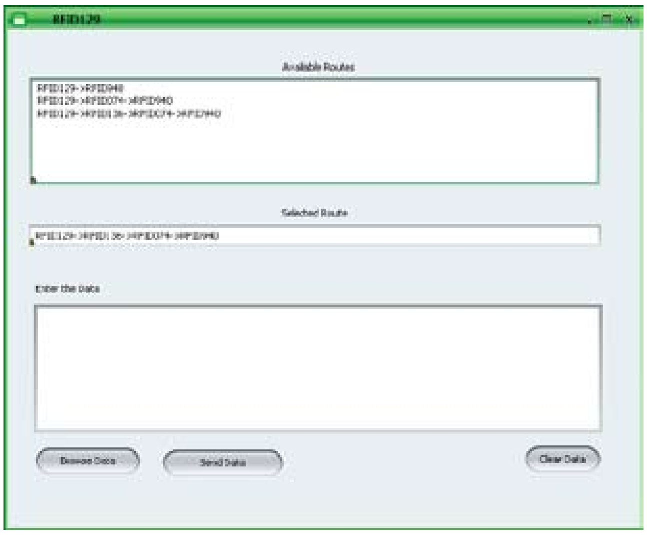
RFID940 server
RFID074 server
RFID136 server
RFID044 server

```
Time In [ RFID044,40,server,7023 ].
Time In [ RFID074,0,server,2720 ].
Time Out [ RFID044,40,server,7023 ].
Time In [ RFID940,45,server,2981 ].
Time In [ RFID136,25,server,8501 ].
Time In [ RFID074,25,server,2720 ].
Time Out [ RFID074,0,server,2720 ]
```

Fig. 4: Movement Graph Status

Fig. 4. Shows the movement status which is calculated by using t_1 (time in) and t_2 (time out), and EPC is calculated by using (EPC; warehouse, time in, time out).

Fig. 5: Shows the available routes and the selected route for sending the data



```
RFID129->RFID940
RFID129->RFID074->RFID940
RFID129->RFID136->RFID074->RFID940
```

RFID129->RFID136->RFID074->RFID940

Fig. 5: Sending Data

IV. Conclusion

In this paper, we have introduced a new, gateway based movement graph model for warehousing massive, and transportation based RFID data sets. This model captures the essential semantics of supply chain application as well as many other RFID applications that explore object movements of similar nature. It provides a clean and concise representation of large RFID data sets. Moreover, it sets up a solid foundation for modeling RFID data and facilitates efficient and effective RFID data compression, data cleaning, multidimensional data aggregation, query processing, and data mining.

In this paper, we have introduced transform time consuming searches into quick, efficient processes by identifying all instances of an item with a single trace. Increase visibility into serial/lot number lifecycles. Conduct powerful searches using an item's serial or lot number. Complete widespread searches across all transactions, including bills of materials and customer orders. Consolidate like lot numbers in all lot number entry windows to gain an accurate view of inventory for a given lot number including manufactured date and expiration date without juggling multiple records.

The gateway based movement graph model proposed here captures the semantics of bulky, sophisticated, but collective object movements, including merging, shuffling, and splitting processes. Its applications are not confined to RFID data sets but can also be extended to other bulky object movement data. However, further study is needed to model enhanced RFID tags using public-key cryptography can help again, since they can respond to a challenge by signing it. The central repository can verify the signature and the freshness of the challenge and then grant access. This could also be a novel way for remote customer service where the customer even remains anonymous.

V. Related Work

RFID Technology has been researched from several perspectives:

1. The physics of building tags and readers [11],
2. The techniques required to guarantee privacy and safety [6,14],
3. The software architecture required to collect, filter, organize, and answer online queries on tags known as the "EPC Global Network," which is defined by several standards including [2,10],
4. Cleaning methods to filter noisy RFID observations [18],
5. Event processing systems on RFID data streams [3], and
6. Storage, warehousing, and mining of massive RFID data sets [8,15-17].

EPC Global Inc. is a standards body that has defined specifications for many of the components of and RFID system. It starts by describing data contained in the tag [12], the communication protocols between reader and tags [16], transmission of raw RFID readings by readers [11]. At a higher level, it specifies how raw readings are converted into events [2], and how such events are stored [10]. Finally, lookup of information about a particular EPC is defined by the object name service [12].

Cleaning of RFID data has been addressed both from the design of robust communication protocols that operate at the hardware level [21,16], and from post processing software designed to detect incorrect readings. At the software level, the use of smoothing windows has been proposed as a method to reduce false negatives. C. Lee and C. Chung, propose the Efficient Storage Scheme and Query Processing for Supply Chain Management Using RFID [12] fixed-window smoothing. Shawn et al. [15] propose a variable-window smoothing that adapts window size dynamically according to tag detection rates, by using a binomial model of tag readings.

Event processing systems is another area of research. RFID systems generate a stream of raw events, which themselves can form higher level events. Work in this area has focused on efficient processing of very large data streams [9-10], definition of complex events, and extensions to standard query languages to account for unique temporal characteristics of RFID events [3], and [5] which cope with noise in the RFID stream by defining probabilistic events over low-level events.

Closer to this work, there is significant research on storage, warehousing, and mining of massive RFID data sets. Gonzalez et al. [17] introduced the problem of compressing and warehousing massive RFID data sets. They proposed the concept of the RFID cuboids which compresses and summarizes an RFID data set by recording information on items that stay together at a location with stay records. This model takes advantage of bulky shipments that are successively split into smaller shipments to compress data. This paper is an extension of [17] to account for a more realistic, item movement model, where items not only split, but can merge and split multiple times as they move through a global supply chain. Lee and Chung [12] propose a very clever path encoding technique that improves on the encoding used in [17] to speed up query processing. At the mining end, Gonzalez et al. [15-16] propose the discovery of workflows from RFID data. Such workflows summarize major flow trends and significant flow exceptions at multiple abstraction levels.

References

[1] S. Chopra and M. Sodhi. "Looking for the Bang from the RFID Buck". Supply Chain Management Review. [Online] Available : http://www.scmr.com/article/CA6444375.html, 2007.

[2] M. Feldhofer, J. Wolkerstorfer, V. Rijmen. "AES Implementation on a Grain of Sand". IEEE Proceedings on Information Security, 152(1):13–20, October 2005.

[3] Santos and L. Smith. "RFID in the Supply Chain: Panacea or Pandora's Box?" Communications of the ACM 51(10), 2008.

[4] T. Plos., "Susceptibility of UHF RFID Tags to Electromagnetic Analysis". In T. Malkin, editor, Topics in Cryptology - CT-RSA 2008, The Cryptographers' Track at the RSA Conference 2008, San Francisco,CA, USA, April 8-11, 2008, Proceedings, volume 4964 of Lecture Notes in Computer Science, pp. 288–300. Springer, April 2008.

[5] B. Santos, L. Smith. "RFID in the Supply Chain: Panacea or Pandora's Box?" Communications of the ACM 51(10), 2008.

[6] J. Strüker, D. Gille, Titus Faupel. "RFID-Report 2008 – Optimizing Business Processes in Germany". IIG-Telematik, Albert-Ludwigs-University Freiburg, VDI Nachrichten, 2008.

[7] L. Weiss Ferreira Chaves, F. Kerschbaum. "Industrial Privacy in RFID-based Batch Recalls". Proceedings of the IEEE International Workshop on Security and Privacy in Enterprise Computing, 2008.

[8] "13.56 MHz ISM Band Class 1 Radio Frequency Identification Tag Interface Specification," technical report, MIT Auto ID Center, 2003.

[9] S.R. Jeffery, M. Garofalakis, M.J. Franklin, "Adaptive Cleaning for RFID Data Streams," Proc. 2006 Int'l Conf. Very Large Data Bases (VLDB '06), Sept. 2006.

[10] S.R. Jeffrey, G. Alonso, M.J. Franklin, W. Hong, J. Widom, "A Pipelined Framework for Online Cleaning of Sensor Data Streams," Proc. 2006 Int'l Conf. Data Eng. (ICDE '06), Apr. 2006.

[11] N. Khoussainova, M. Balazinska, D. Suciu, "Peex: Extracting Probabilistic Events from RFID Data," Proc. 2008 Int'l Conf.

Data Eng. (ICDE '08), pp. 1480-1482, Apr. 2008.
[12] C. Lee, C. Chung, "Efficient Storage Scheme and Query Processing for Supply Chain Management Using RFID," Proc. 2008 ACM Special Interest Group on Management of Data Int'l Conf. Management of Data (SIGMOD '08), pp. 291-302, June 2008.
[13] EPCglobal Object Name Service (ONS) 1.0.1, Standard, EPCglobal, [Online] Available : http://www.epcglobalinc.org/standards/ons, 2008.
[14] J. Rao, S. Doraiswamy, H. Thakar, L.S. Colby, "A Deferred Cleansing Method for RFID Data Analytics," Proc. 2006 Int'l Conf. Very Large Data Bases (VLDB '06), Sept. 2006.
[15] Reader Protocol (RP) Standard, Standard, EPCglobal, [Online] Available : http:// www.epcglobalinc.org/standards/rp, 2006.
[16] S.E. Sarma, S.A. Weis, D.W. Engels, "RFID Systems and Security and Privacy Implications," Proc. Workshop Cryptographic Hardware and Embedded Systems, pp. 454-470, 2002.
[17] K. Muralidhar, R. Sarathy, "A General Additive Data Perturbation Method for Database Security," Management Sciences, vol. 45, pp. 1399-1415, 1999.
[18] E. Bertino, R. Sandhu, "Database Security: Concepts, Approaches, and Challenges," IEEE Trans. Dependable and Secure Computing, vol. 2, no. 1, pp. 2-19, Jan.-Mar. 2005.

P. Mohana Priya received the BCA and MCA degrees from Bharathiar University, Coimbatore, in 2005 and 2008, respectively. She is currently working toward the ME degree in Computer Science and Engineering in Karpagam University. Her research interests include data security, data mining, warehousing and Steganography.

B. Sangeetha received the Bsc, Msc and M.Phil degrees from Bharathiar University, Coimbatore, in 2004, 2006 and 2008 respectively. She is currently working as a lecturer in Department of Computer Application in Karpagam University. Her research interests include data security, data mining, Advanced Networks.

V.G. Vijaya Kumaran received the BCA, Msc and M.Phil degrees from Bharathiar University, Coimbatore, in 2004, 2006 and 2008 respectively. She is currently working as a lecturer in Department of Computer Application in Karpagam University. Her research interests include data security, data mining, Advanced Networks.