International Journal of Engineering Innovation & Research
Volume 2, Issue 6, ISSN: 2277 – 5668

Secured Query Processing in Wireless Sensor Networks

Dr. Keshava Prasanna
Associate Professor, Deptt. of CSE/ISE , BMSIT, Bangalore
Email: keshava2011@rediffmail.com

Dr. Thungamani M.
Associate Professor, Deptt. of CSE/ISE , BMSIT, Bangalore
Email: thungamani_k@rediffmail.com

***Abstract* – Sensing and Wireless technology has evolved to the point where hundreds of tiny, battery-powered sensors are deployed for monitoring applications that continuously collect data about its surroundings. These sensors need to process the data which it collects into data streams to be fed into a database, and this is where network connecting the sensors comes into play. The very aspect of mobility and low setup cost makes wireless sensor networks easier to implement than the wired ones.**

The query processing in wireless sensor networks deals with the query and information retrievation of data to and from the sensor nodes to the user ends. Now as we know, we have a network and routing layer that allow us to communicate with any node in the network, we will look at how data streams or tuples are processed within the network. Each data stream in the network is labeled with a unique identifier. Data streams are created by the actual sensors collecting data and by operators in the network. The query processor module is responsible for delivering the tuples to the correct operator and node. The paper discusses all these processing along with the architecture of the query processor.

***Keywords* – Network Topology, Wireless Sensor Networks, Query Processing, Query Optimization, Epoch.**

I. INTRODUCTION

1.1 Wireless Sensor Networks

A **wireless sensor network** (WSN) is a network made of numerous small independent **sensor** nodes.

The sensor nodes, typically the size of a 35 mm film canister, are self-contained units consisting of a **battery**, radio, sensors, and a minimal amount of on-board computing power. The nodes self-organize their networks [1], rather than having a pre-programmed **network topology**. Because of the limited electrical power available, nodes are built with power conservation in mind, and generally spend large amounts of time in a low-power "sleep" mode. The underlying hardware technology for wireless sensor networks, consisting of perhaps thousands of integrated devices, with built- in processing, storage and sensors with RF transceiver, energy storage and antenna is evolving quickly and gaining a signature style of design. That design involves many energy constrained, resource-limited devices operating in concert as a result of application requirements demanding long-term operation, up-close monitoring and constraints on size and available power.

1.2 Applications of Wireless Sensor Networks

These sensors can be used in a wide range of different **monitoring** and **data retrieval tasks (Query Processing)**. These can be roughly classified into

- Monitoring space,
- Monitoring things, and
- Monitoring the interactions of things with
- Each other and the encompassing space.

The first category includes environmental and habitat monitoring, precision agriculture, indoor climate control, surveillance, treaty verification, and intelligent alarms [2]. The second includes structural monitoring, condition-based equipment maintenance, medical diagnostics, and urban terrain mapping. The most dramatic applications involve monitoring complex interactions, including wildlife habitats, disaster management, emergency response, ubiquitous computing environments, asset tracking, healthcare, and manufacturing process flow.

II. QUERY PROCESSING

2.1. Query Processing Architecture

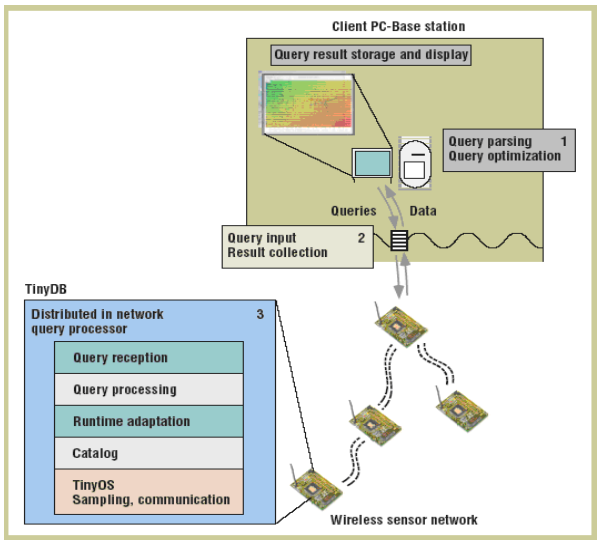


Fig.2.1. Architecture of a wireless sensor network query processor. Numbers indicate the sequence of steps involved in processing a query.

We have a network and routing layer that allows us to communicate with any node in the network; now we look at how data streams or tuples are processed within the network. Each data stream in the network is labeled with **a unique identifier**. Data streams are created by the actual sensors collecting data and by operators in the network; tuples travel along these data streams from operator to operator. The query processor module is responsible for delivering the tuples to the correct operator and node[3]. When the query is first sent out to the sensor network, each node evaluates the query and identifies the operations that it should execute as specified by the base station. **Figure 2.1** shows a simple block diagram of architecture for query processing in wireless sensor networks.

The architecture has two main parts:

1. Server-side software running on the user's PC—the **base station**. In its most basic form, the software parses queries, delivers them into the network, and collects results as they stream out of the network.

2. Sensor-side software running on the motes. As the "Distributed in network query processor" detail box in Figure shows, this software consists of several components built on top of TinyOS. One of the motes—the **root**-communicates with the base station.

2.2 Need for Query Processing In Wireless Sensor Networks

Wireless Sensor Networks produce data in continuous streams; often at defined intervals without having been explicitly prompted for the data. Because of the limited hardware, they can not store a lot of the data they collect locally. These sensors usually send their data to a [4] workstation where the data can be collected, stored and processed. The sensor network communicates with the workstation (base station) through a gateway sensor which is connected to the base station. Gateway sensor ultimately is the root of the sensor network because this is where all of the data streams must go through to communicate with the base station.

2.3 Introducing Query Langu -Age And Query Optimization

In the architecture, users input queries at the server in a simple, **SQL-like language** that describes the data they wish to collect and how they wish to combine, transform, and summarize it. The SQL variant differs most significantly from traditional SQL in that its queries are continuous and periodic. That is, users register an interest in certain kinds of

Sensor readings [5] (for example, "temperatures from sensors on the fourth floor every five seconds"), and the system streams the results to the user. Each period in which a result is produced is an **epoch**. The epoch duration, or sample period of a query, refers to the amount of time between successive samples (five seconds in this example). In this paper, we consider queries of the simple form shown in **Figure 2.3a** below.

```
SELECT      {attributes, aggregates}
FROM        {Sensordata S}
WHERE       {predicate}
GROUP BY    {attributes}
HAVING      {predicate}
DURATION    time interval
EVERY       time span e
```

Fig.2.3a. Basic Query Block

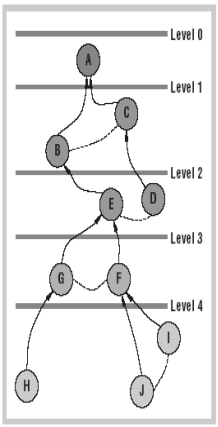
The process of selecting the best possible plan is called **query optimization**. In sensor networks, query optimization, because it can be computationally intensive, occurs as much as possible on the server-side PC. **For example**, to monitor whether the average concentration of a chemical is above a certain threshold, we can use the long-running query shown in **Figure 2.3b**, but there is a delay of 10 seconds between every recomputation of the average.

```
SELECT      AVG(R.concentration)
FROM        ChemicalSensor R
WHERE       R.loc IN region
HAVING      AVG(R.concentration) > T
DURATION    (now,now+3600)
EVERY       10
```

Fig.2.3b. Example Query

2.4 Query Dissemination and Result Collection

After optimizing a query, the system disseminates it into the network. A routing tree is a communication primitive rooted at either the base station or a storage point. The routing tree is formed as nodes forward the query to other nodes in the network: The network root transmits the query, and all child nodes hearing the query, process it and forward it to their children, and so on until the entire network has heard the query[6] . The parent will be responsible for forwarding the node's (and its children's) query results to the base station. If nodes keep track of multiple parents, the network can have several routing trees, which can support several simultaneous queries with different roots. This type of communication topology, known as **tree-based routing**, is common within the sensor network community.



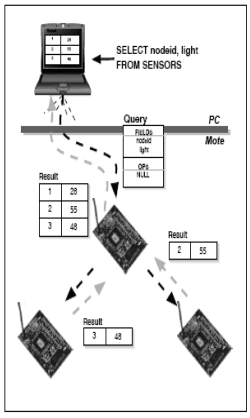


Fig.2.4a. Sensor network topology

Fig. 2.4b. A query and result propagation with routing tree overlay.

Figure 2.4a shows an example sensor network topology and routing tree. Solid arrows indicate parent nodes, and dotted lines indicate nodes that can hear each other but don't route through each other. A node can generally choose a parent from several possible nodes [6] ; a simple approach is to choose the ancestor node at the lowest level. In practice, choosing the proper parent is quite important in terms of communication and data collection efficiency. Other than routing trees because the overlay of several routing trees neglects any sharing between trees and leads to performance decay. Because the wireless sensor network is programmed only through queries, regular

communication patterns exist, mainly consisting of the collection of sensor readings from a region at a single node or the base station. A query workload with more than a few destinations requires routing structures other than routing trees because the overlay of several routing trees neglects any sharing between trees [7] and leads to performance decay. Similarly, the **Figure 2.4b** shows how a **query and its result propagate** through the wireless sensor networks.

2.5 Types of Queries in Wireless Sensor Networks

• *Monitoring Queries:* Queries that request the value of one or more attributes continuously and periodically – for example, reporting the temperature in bird nests every thirty seconds; these are similar to the query shown above.

• *Network Health Queries:* Meta-queries over the network itself. Examples include selecting parents and neighbors in the network topology or nodes with battery life less than some threshold. These queries are particularly important in sensor networks due to their dynamic and volatile nature. **For example**, the following query reports all sensors whose current battery voltage is less than k:

```
SELECT nodeid,voltage
  WHERE voltage < k
  FROM sensors
  SAMPLE PERIOD 10 minutes
```

• *Exploratory Queries:* One-shot queries examining the status of a particular node or set of nodes at a point in time. In lieu of the **SAMPLE PERIOD** clause, users may specify the keyword **ONCE**. For example:

```
SELECT light,temp,volume
  WHERE nodeid = 5
  FROM sensors
  ONCE
```

• *Actuation Queries:* Users want to able to take some physical action in response to a query. We include a special **OUTPUT ACTION** clause for this purpose. For example, users in building monitoring scenarios might want to turn on a fan in response to temperature rising above some level:

```
SELECT nodeid,temp
  FROM sensors
  WHERE temp > threshold
  OUTPUT ACTION power-on(nodeid)
  SAMPLE PERIOD 10s
```

The **OUTPUT ACTION** clause specifies an external command that should be invoked in response to a tuple satisfying the query. In this case, the power-on command is a low-level piece of code that pulls an output pin on the microprocessor high, closing a relay circuit and giving power to some externally connected device. The **OUTPUT ACTION** suppresses the delivery of messages to the base station.

2.6 Working of Query Processing in WSN

After a query has been disseminated, each node begins processing it. Processing is a simple loop: once per epoch, a special acquisition operator at each node acquires readings, or samples, from sensors corresponding to the fields or attributes referenced in the query. The query processor routes this set of readings, or tuple, through the query plan built in the optimization phase [8]. The plan consists of number of operators applied in a fixed order. Each operator can pass the tuple to the next operator, reject it, or combine it with one or more other tuples. A node transmits tuples that successfully pass the plan up the routing tree to the node's parent, which can, in turn, forward the result or combine it with its own data or data collected from other children.

Table 2.5: Common wireless sensor network query-processing operators

Operator	**Description**
Data Acquisition	Acquire a reading (field) from a sensor or an internal device attribute, such as a light sensor reading or free RAM in the dynamic heap.
Select	Reject readings that don't satisfy a particular Boolean predicate. For example, the predicate temp > 80°F rejects readings under 80°F.
Aggregate	Combine readings according to an aggregation function. For example, AVG (light) computes the average light value over each mote.
Join	Concatenate two readings when some join predicate is satisfied.

Table 2.5 describes some common query processing operators used in WSNQPs. The data acquisition operator uses a catalog of available attributes to map names referenced in queries to low-level operating system functions that can be invoked to provide their values. This catalog abstraction lets sophisticated users extend the sensor network with new kinds of sensors and provides support for sensors accessed via different software interfaces. Figure 2.5 illustrates query processing for the simple aggregate query, "Tell me the average temperature on the fourth floor once every five seconds". Here, the query plan contains three operators: a data acquisition operator, a select operator that checks whether the value of the floor attribute equals 4 and an aggregate operator that computes the temperature attribute average from the local mote and the average temperature values of any of the mote's descendants that are on the fourth floor. Each sensor applies this plan once per epoch, and the data stream.

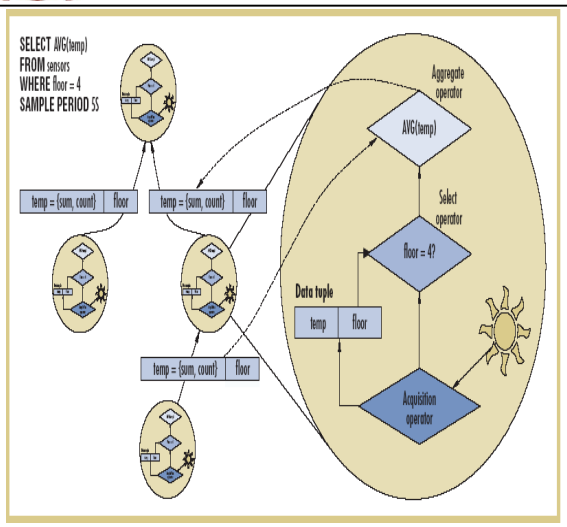


Fig.2.5. Sensor network executing a simple aggregate query.

Produced at the root node is the answer to the query. We represent the partial computation of averages as {sum, count} pairs, which are merged at each intermediate node in the query plan to compute a running average as data flows up the tree. Finally, for this example, if the nodes are known to be immobile, the predicate over floor will be constant valued, meaning that nodes on floors other than the fourth will never produce values for the query. TinyDB includes the notion of a **semantic routing tree**, 4 which lets nodes opt out of queries with nonsatisfiable predicates over constant-valued attributes [9]. However, such nodes might still have to forward packets on behalf of other nodes that satisfy the predicate.

III. Security Regarding Query Processing in WSN

3.1 Why Security is Necessary in Query Processing

- To make wireless sensor networks economically viable, and query processing effective is a challenge, as the sensor devices are limited in their energy, computation and communication capabilities, thus eliminating the scope of Artificial Intelligence in them.
- Unlike traditional networks, sensor nodes are often deployed in accessible areas, presenting the added risk of physical attack.
- Sensor networks and query processing interact closely with their physical environments and with people, posing new security problems.
- As the sensor nodes [10] are architecturally simple, so they can easily be duplicated to intercept the data streams from query processors and vice-versa.

3.2 Problems and Solutions Related to Secured Query Processing in WSN

3.2.1 When setting up a sensor network and query processor, one of the first requirements is to established **cryptographic keys** for later use. **Key establishment techniques need to scale to networks with hundreds or thousands of nodes**. Moreover, the communication patterns of sensor networks differ from traditional networks; sensor nodes need to setup keys with their neighbors and with data aggregations nodes.

- **The simplest solution for key establishment** is a network wide shared key. Unfortunately, the compromise of even a single node in a network would reveal the secret key and thus allow decryption of all network traffic regarding query processing. However this variant of the key establishment process doesn't allow addition of new nodes after initial deployment.
- **Public-key cryptography is another option** beyond the capabilities of today's sensor networks. Its main advantage is that a node can setup a secure key with any other node in the network.
- **Bootstrapping keys using a trusted base station is another option**. Here, each node needs to share only a single key with the base station and setup keys with other nodes through the base station. This arrangement makes the base station a single point of failure, but because there is only one base station, the network may incorporate temper-resistant packaging for the base station, ameliorating the threat of physical attack.

3.2.2 Like traditional networks, the **data flow phenomena in query processing requires protection against eavesdropping, injection and modification of packets.**

- **Cryptography is the standard defense**. Interesting system trade-offs arise when incorporating cryptography into sensor networks. For point to point communication, end to end cryptography achieves a high level of security but requires that keys be setup among all end points.
- **Link-layer cryptography with network wide shared key** simplifies key setup and supports passive participation and local broadcast, but intermediate nodes might eavesdrop or alter messages.
- **Cryptography entails a performance for extra computation that often increases packet size**. Cryptographic hardware increases efficiency but also increases the financial cost of implementing a network. Thus, an important question facing query processor researchers is: can reasonable security and performance levels be achieved with software-only cryptographic implementation, or is hardware support needed?
- Recent research shows that **software only cryptography** is indeed practical with today's sensor and query technology; hardware support is not needed to
- Achieve acceptable security and performance levels.

3.2.3 Query processing in wireless sensor networks has also **thrust privacy concerns to the forefront.** The most obvious risk is that ubiquitous sensor technology might allow individuals to deploy secrets surveillance networks for spying on unaware victims. Employers might spy on their employees, shop owners might spy on customers, neighbors might spy on each other and law enforcement agencies might spy on public places. .**Providing awareness of presence of sensor nodes date acquisition is particularly important.**

Technology alone is unlikely to be able to solve problems regarding privacy; rather, **a mix of societal**

norms, new laws and technological responsibilities are necessary.

3.2.4 **Adversaries can severely limit the value of wireless sensor network through denial-of-service attacks**. In its simplest form, an adversary attempts to disrupt the query processor and the networks operation by broadcasting a high energy signal. If the transmission is powerful enough the **entire systems communication could be jammed**. More sophisticated attacks are also possible, the adversary might inhibit communication by violating the **802.11 medium access control protocol** by continuously requesting channel access with a request-to-send signal.

One standard defense against jamming employs spread-spectrum communication. However, cryptographically secure spread-spectrum radios are not commercially available. Also this defense is not secure against adversaries who might capture nodes and extracts their cryptographic keys.

The network nature of query processing allows new, **automated defenses against denial of service.** When the jamming affects only a portion of the network, a jamming-resistant network could defeat the attack by detecting the jamming, mapping the affected region, and then routing around the jammed area.

IV. CONCLUSION

We have presented a system that allows processing of a query on a sensor network platform. By splitting up a query posed to the sensor network as a tree of operators and distributing these operators onto the nodes of the network, the system has more flexibility in placing operators within the network and can more efficiently dictate the data flow within the network.

Processing all of the data produced by the entire sensor network is highly inefficient because of the energy consumption difference in using the radio and using the processor. Thus we took the database perspective and viewed the sensor network as a special distributed database where queries would be executed on the data streams coming from these sensors.

Secured query processing in WSN forms a very specific field of WSN, which deals with various security concerns and their possible solutions. Among them secure routing is perhaps most vital to the acceptance and use of sensor networks for many applications, but we have discussed that currently proposed routing protocols for these networks are insecure. Link layer encryption and authentication mechanisms may be a reasonable first approximation for defense against remote-class outsiders, but cryptography alone is not enough. The possible presence of laptop-class adversaries and insiders and the limited applicability of end- to- end security mechanisms necessitates careful protocol design as well.

REFERENCES

[1] B. Chen, K. Jamieson, H. Balakrishnan, R. Morris, Span: an energy-efficient coordination algorithm for topology maintenance in ad hoc wireless networks, ACM Wireless Networks Journal 8 (5) (2002) 481–494.

[2] Y.-C. Hu, A. Perrig, D.B. Johnson, Packet leashes: a defense against wormhole attacks in wireless networks, in: IEEE Infocom, 2003

[3] C. Intanagonwiwat, R. Govindan and D. Estrin, "Directed diffusion: A scalable and robust communication paradigm for sensor networks", in the *Proceedings of the 6th Annual ACM IEEEInternational Conference on Mobile Computing and Networking (MobiCom'00)*, Boston, MA, August 2000.

[4] D. Estrin, et al., "Next century challenges: Scalable Coordination in Sensor Networks," in the *Proceedings of the 5th annual ACM/IEEE international conference on Mobile Computing and Networking (MobiCom'99)*, Seattle, WA, August 1999.

[5] B. Krishnamachari, D. Estrin, S. Wicker, "Modeling Data Centric Routing in Wireless Sensor Networks," in the *Proceedings of IEEE INFOCOM*, New York, NY, June 2002.

[6] D. Braginsky, D. Estrin, Rumour routing algorithm for sensor networks, in: First ACM International Workshop on Wireless Sensor Networks and Applications, 2002.

[7] L. Li, J. Halpern, Z. Haas, Gossip-based ad hoc routing, in: IEEE Infocom 2002, 2002.

[8] C. Intanagonwiwat, R. Govindan, D. Estrin, Directed diffusion: a scalable and robust communication paradigm for sensor networks, in: Proceedings of the Sixth Annual International Conference on Mobile Computing and Networks(Mobi-COM _00), 2000.

[9] F. Ye, A. Chen, S. Lu, L. Zhang, A scalable solution to minimum cost forwarding in large sensor networks, in: Tenth International Conference on Computer Communications and Networks, 2001, pp. 304–309.

[10] D. Braginsky and D. Estrin, "Rumor Routing Algor ithm for Sensor Networks,"in the *Proceedings of the First workshop on Sensor Networks and Applications (WSNA)*, Atlanta,GA,October2002.

AUTHOR'S PROFILE

Keshava Prasanna
completed his Doctorate from Tumkur University, working as a Associate Professor in Department of Computer Science And Engineering, BMS Institute of Technology, Bangalore.

Thungamani. M
completed her Doctorate from Tumkur University, working as a Associate Professor in Department of Computer Science And Engineering, BMS Institute of Technology, Bangalore.