Data Integrity and Delay Differentiated Services in Wireless Sensor Networks Using Dynamic Routing

[1]S Mahesh Kumar
[1]Asst. Professor
[1]Department Of CSE
[1]KG Reddy College of Engineering and Technology, Hyderabad, Telangana

Abstract: With the enormous advancement in the field of embedded computer and sensor technology, Wireless Sensor Networks (WSNs) have made remarkable impact in today's world. These WSNs consist of several thousands of sensor nodes deployed randomly, are capable of sensing, actuating, and communicating the collected information. Since wireless sensor networks are constrained by cost, scalability, topology change and power consumption, new technologies are being considered to overcome these and many other issues. Applications running on the same Wireless Sensor Network (WSN) platform usually have different Quality of Service (QoS) requirements. Two basic requirements are low delay and high data integrity. However, in most situations, these two requirements cannot be satisfied simultaneously. In this paper, based on the concept of potential in physics, we propose IDDR, a multi-path dynamic routing algorithm, to resolve this conflict. By constructing a virtual hybrid potential field, IDDR separates packets of applications with different QoS requirements according to the weight assigned to each packet, and routes them towards the sink through different paths to improve the data fidelity for integrity-sensitive applications as well as reduce the end-to-end delay for delay-sensitive ones. Using the Lyapunov drift technique, we prove that IDDR is stable. Simulation results demonstrate that IDDR provides data integrity and delay differentiated services.

Keywords: Accuracy, Prediction algorithms, Training, Partitioning algorithms, Stability criteria, Estimation, Redundancy.

I. INTRODUCTION

Wireless sensor networks (WSNs) have gained worldwide attention in recent years, particularly with the proliferation in Micro-Electro-Mechanical Systems (MEMS) technology which has facilitated the development of smart sensors. These sensors are small, with limited processing and computing resources, and they are inexpensive compared to traditional sensors. These sensor nodes can sense, measure, and gather information from the environment and, based on some local decision process, they can transmit the sensed data to the user. Smart sensor nodes are low power devices equipped with one or more sensors, a processor, memory, power supply, radio, and an actuator. WSNS, which are used to sense the physical world, will play an important role in the next generation networks. Due to the diversity and complexity of applications running over WSNs, the QoS guarantee in such networks gains increasing attention in the research community. As a part of an information infrastructure, WSNs should be able to support various applications over the same platform. Different applications might have different QoS requirements. For instance, in a fire monitoring application, the event of a fire alarm should be reported to the sink as soon as possible. On the other hand, some applications require most of their packets to successfully arrive at the sink irrespective of when they arrive. For example, in habitat monitoring applications, the arrival of packets is allowed to have a delay, but the sink should receive most of the packets. WSNs have two basic QoS requirements: low delay and high data integrity, leading to what are called delay sensitive applications and high-integrity applications, respectively. Generally, in a network with light load, both requirements can be readily satisfied. However, a heavily loaded network will suffer congestion, which increases the end-to-end delay. This work aims to simultaneously improve the fidelity for high-integrity applications and decrease the end-to-end delay for delay-sensitive ones, even when the network is congested. We borrow the concept of potential field from the discipline of physics and design a novel potential based routing algorithm, which is called integrity and delay differentiated routing (IDDR). IDDR is able to provide the following two functions:

Improve Fidelity for High-Integrity Applications: The basic idea is to find as much buffer space as possible from the idle and/or under-loaded paths to cache the excessive packets that might be dropped on the shortest path. Therefore, the first task is to find these idle and/or under loaded paths, then the second task is to cache the packets efficiently for subsequent transmission. IDDR constructs a potential field according to the depth1 and queue length information to find the under-utilized paths. The packets with high integrity requirement will be forwarded to the next hop with smaller queue length. A mechanism called Implicit Hop-by-Hop Rate Control is
designed to make packet caching more efficient.

Decrease End-To-End Delay for Delay-Sensitive Applications: Each application is assigned a weight which represents the degree of sensitivity to the delay Through building local dynamic potential fields with different slopes according to the weight values carried by packets, IDDR allows the packets with larger weight to choose shorter paths. In addition, IDDR also employs the priority queue to further decrease the queuing delay of delay sensitive packets. IDDR inherently avoids the conflict between high integrity and low delay: the high-integrity packets are cached on the under loaded paths along which packets will suffer large end-to-end delay because of more hops, and the delay- sensitive packets travel along shorter paths to approach the sink as soon as possible. Using the Lyapunov drift theory, we prove that IDDR is stable. Furthermore, the results of a series of simulations conducted on the TOSSIM platform demonstrate the efficiency and feasibility of the IDDR scheme.

II. Literature survey

Applications running on the same Wireless Sensor Network (WSN) platform usually have different Quality of Service (QoS) requirements. Two basic requirements are low delay and high data integrity. However, in most situations, these two requirements cannot be satisfied simultaneously. In this paper, based on the concept of potential in physics, we propose IDDR, a multi-path dynamic routing algorithm, to resolve this conflict. By constructing a virtual hybrid potential field, IDDR separates packets of applications with different QoS requirements according to the weight assigned to each packet, and routes them towards the sink through different paths to improve the data fidelity for integrity-sensitive applications as well as reduce the end-to-end delay for delay-sensitive ones. Using the Lyapunov drift technique, we prove that IDDR is stable. Simulation results demonstrate that IDDR provides data integrity and delay differentiated services.

Receiver's prospects:

Improve fidelity for high-integrity applications. The basic idea is to find as much buffer space as possible from the idle and/or under-loaded paths to cache the excessive packets that might be dropped on the shortest path. Therefore, the first task is to find these idle and/or underloaded paths, then the second task is to cache the packets efficiently for subsequent transmission. IDDR constructs a potential field according to the depth1 and queue length information to find the under-utilized paths. The packets with high integrity requirement will be forwarded to the next hop with smaller queue length. A mechanism called Implicit Hop-by-Hop Rate Control is designed to make packet caching more efficient.

Steganography technique in which the sender sends a stego - image to the receiver or legitimate user. This user having the stego key to extract secret data from stego image. The legitimate user must have the same key with which the image is embedded. On stego image extracting process is applied b using patching techniques and exchange alteration techniques. Finally we get the secret data which is embedded.

Hash function algorithm

A hashing algorithm creates a hash code, also called a "message digest" or "message fingerprint". Hash codes are of limited use for communications security, because sender can replace both the hash code and the message received by the receiver, but they are an essential element of digital signatures for sharing keys between the sender and receiver.
In this section, we explain how hashing algorithms work, and provide some practical insight into choosing a suitable algorithm for the proposal.

Creating a hash code

At the heart of a hashing algorithm is a mathematical function that operates on two fixed-size blocks of data to create a hash code as shown in the below figure.

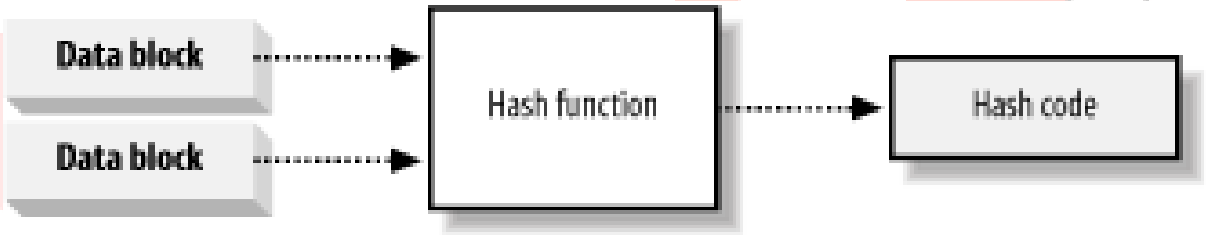


Figure -1: The hash function operates on fixed size blocks of data.

It break ups senders message into blocks that are the same size as the input for the hash function. The size of each data block varies depending on the algorithm but the blocks tend to be small the algorithms included in the framework break the messages into blocks of 512 or 1024 bits. The design of the hash functions for each hashing algorithm, but they all share the same basic approach.

The algorithm specifies a "seed" value that feeds into the hash function along with the first block of message data, thus producing our first hash code. Take this hash code and feed it into the hash function along with the second block of message data, creating a second hash Code. Feed the second hash code into the function along with the third message block, and repeat the process of hashing a data block along with the third message block, and repeat the process of hashing a data block along with the hash code of the previous block until processing of all the message data as shown in the figure.

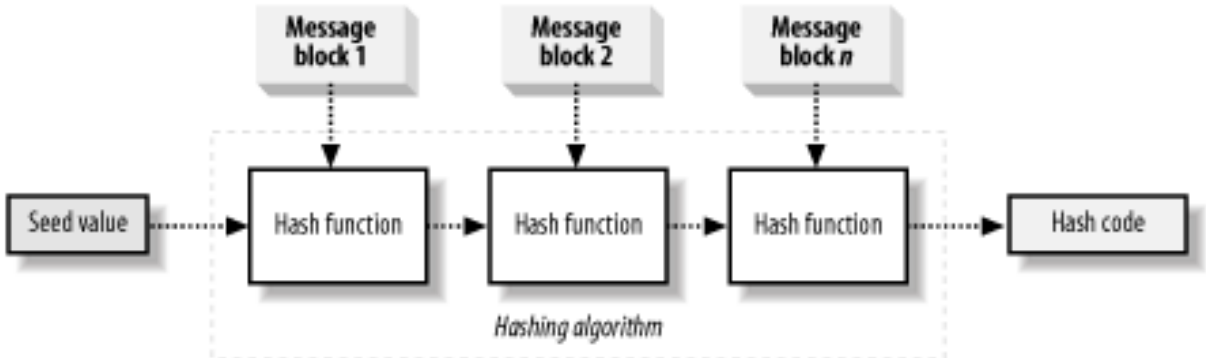


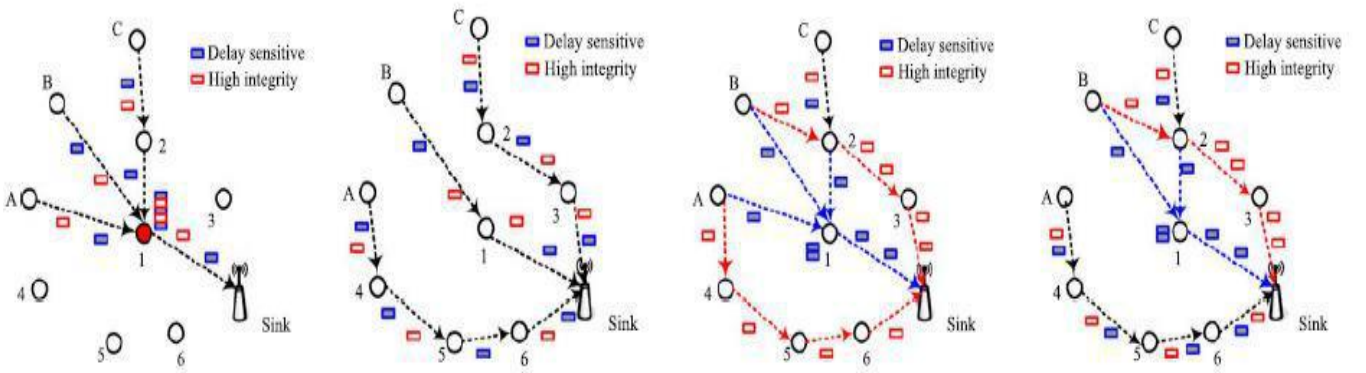
Figure 2 : Creating the hash code by "chaining" the hash function.

The idea of a hash code acting as a message "fingerprint" is reasonable, because making even the smallest change in the message data changes the value of the final hash code. By "chaining" repeated calls to the hashing function, you can create a hash code that relies on the value of every single bit of the message. The output of hashing the first data block becomes an input to the next operation, which will alter the value of the second hash code, which will affect the result of the third operation, and so on. Known as "ripple effect" or an "avalanche" the huge impact that the smallest change has on the final hash code provides the fingerprint. Two messages that differ by a single bit of data produce very different hash codes.

III. Proposed system

Mechanism of proposed technique is as follows

Sender's prospects



Architecture

A. Service Provider

In this module, the service provider will browse the data file, initialize the router nodes and then send to the particular receivers. Service provider will send their data file to router and router will select smallest distance path and send to particular receiver.

B. Router

The Router manages a multiple networks to provide data storage service. In network n-number of nodes are present (n1, n2, n3, n4, n5…). In a router service provider can view node details and attacked nodes. Service provider will send their data file to router and router will select smallest distance path and send to particular receiver. If any attacker is found in a node then router will connect to another node and send to particular user.

C. IDS Manager

In this module, the IDS Controller consists of two phases. If Integrity or Malicious Data is occurs in router then IDS controller is activated. In a first phase DNS packets, Net flow, Traffic filter and Fine-grained IDS client detection are present. Aim is that detecting all hosts within the monitored network that engage in IDS communications. We analyze raw traffic collected at the edge of the monitored network and apply a pre-filtering step to discard network flows that are unlikely to be generated by IDS applications. We then analyze the remaining traffic and extract a number of statistical features to identify flows generated by IDS clients. In the second phase, Coarse-grained IDS Integrity or Malicious Data detection, Fine-grained IDS client detection and Integrity or Malicious Data are present; our system analyzes the traffic generated by the IDS clients and classifies them into either legitimate IDS clients or IDS Integrity or Malicious Data.

D. Receiver (End User)

In this module, the receiver can receive the data file from the router. Service provider will send data file to router and router will send to particular receiver. The receivers receive the file by without changing the File Contents. Users may receive particular data files within the network only.

E. Attacker

Attacker is one who is injecting malicious data to the corresponding node and also attacker will change the bandwidth of the particular node. The attacker can inject fake bandwidth to the particular node. After attacking the nodes, bandwidth will have changed in a router.

Receiver's prospects:

To evaluate the performance of IDDR in large-scale WSNs, a series of simulations are conducted on the TOSSIM platform built in TinyOS [2]. We specify IDDR with α = 0.6 and α =0.4 as the representative algorithms. As previously stated after the remarks of Proposition 1 and 2,

when α = 0.4, the packets are allowed to be sent to the neighbors with the same or higher depth to bypass the hotspots, whereas when α= 0.6, the packets are only allowed to be transmitted to the neighbors with the same depth. The performance of IDDR with α = 0.6 and α = 0.4 is compared with Mint Route and IDDR with α = 0.1.

Fig. 2 shows a randomly deployed rectangular network and three monitoring areas. 600 nodes spreading over a 100×100 meters square form a flat multi-hop network. There is only one sink residing at the center, and the communication range is 6 meters. The detailed deployment configuration is summarized in Table 1. There are three applications running over the network from 100 s to 160 s: APP 1 is one high-integrity application generating packets with weight of 0 at the sampling rate of 4 Kbps. APP 2 and APP 3 are delay-sensitive applications and generate packets with weights of 50 and 200, respectively at the sampling rate of 8 Kbps. Fig. 3 indicates the positions of the monitoring areas and Table 2 describes when and how routing is done.

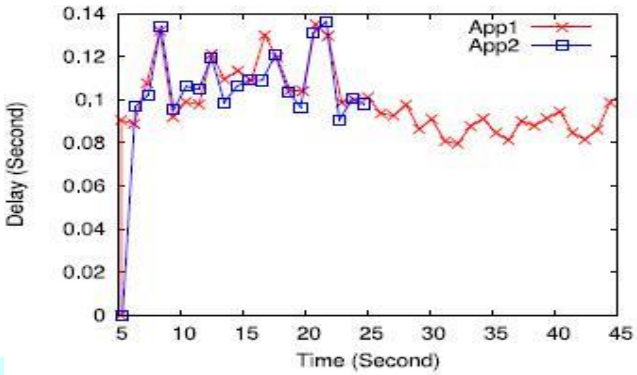


Fig.2. Average Packet Delay of Each Application Under Mint Route on the Test Bed.

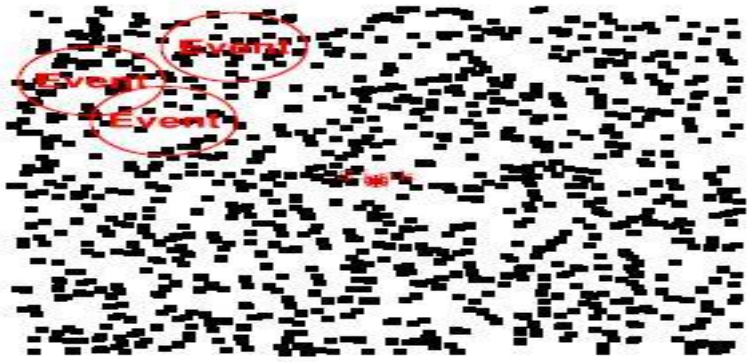


Fig.3. Simulation Topology: Randomly Deployed Network.

These results indicate IDDR significantly improves the throughput. Fig. 4 presents the receiving packet rate, i.e., the rate at which the sink receives packets, which explains why IDDRs have higher throughput. A lot of packets that are likely dropped under other routing algorithms are cached for a short time and eventually reach the sink in IDDR. Thus, IDDR successfully smooth's the bursts and prevents the burst packets from dropping. On the contrary, Mint Route drops most of the burst packets. Although the shortest path algorithm (IDDR with α = 1.0) performs much better than Mint Route, both of them receive fewer packets at the sink out of the bursting time (100s ~130s), while IDDRs continue to receive a lot. Rule 1 plays an important role in improving the throughput.

IV. Conclusion

In this paper, a dynamic multipath routing algorithm IDDR is proposed based on the concept of potential in physics to satisfy the two different QoS requirements, high data fidelity and low end-to-end delay, over the same WSN simultaneously. The IDDR algorithm is proved stable using the Lyapunov drift theory. Moreover, the experiment results on a small test bed and the simulation results on TOSSIM demonstrate that IDDR can significantly improve the throughput of the high-integrity applications and decrease the end-to-end delay of delay sensitive applications through scattering different packets from different applications spatially and temporally. IDDR can also provide good scalability because only local information is required, which simplifies the implementation. In addition, IDDR has acceptable communication overhead.

V. References

1. [1]Jiao Zhang, Member, IEEE, Fengyuan Ren, Member, IEEE, Shan Gao, Hongkun Yang, and Chuang Lin, Senior Member, IEEE, "Dynamic Routing for Data Integrity and Delay Differentiated Services in Wireless Sensor Networks", IEEE Transactions on Mobile Computing, Vol. 14, No. 2, February 2015.
2. [2]P. Levis, N. Lee, M. Welsh, and D. Culler, "TOSSIM: Accurate and scalable simulation of entire TinyOS applications," in Proc. 1st Int. Conf. Embedded Networked Sensor Syst., 2003, pp. 126–137.
3. [3]T. Chen, J. Tsai, and M. Gerla, "QoS routing performance in multihop multimedia wireless networks," in Proc. IEEE Int. Conf. Universal Personal Commun., 1997, pp. 557–561.
4. [4]R. Sivakumar, P. Sinha, and V. Bharghavan, "CEDAR: Core extraction distributed ad hoc routing algorithm," IEEE J. Selected Areas Commun., vol. 17, no. 8, pp. 1454–1465, Aug. 1999.

5. [5]S. Chen and K. Nahrstedt, "Distributed quality-of-service routing in ad hoc networks," IEEE J. Selected Areas Commun., vol. 17, no. 8, pp. 1488–1505, Aug. 1999.
6. [6]B. Hughes and V. Cahill, "Achieving real-time guarantees in mobile ad hoc wireless networks," in Proc. IEEE Real-Time Syst. Symp., 2003.
7. [7]E. Felemban, C.-G. Lee, and E. Ekici, "MMSPEED: Multipath multi-speed protocol for QoS guarantee of reliability and timeliness in wireless sensor networks," IEEE Trans. Mobile Comput., vol. 5, no. 6, pp. 738–754, Jun. 2003. [8]C. Lu, B. Blum, T. Abdelzaher, J. Stankovic, and T. He, "RAP: A real-time communication architecture for large-scale wireless sensor networks," in Proc. IEEE 8th Real-Time Embedded Technol. Appl. Symp., 2002, pp. 55–66.
8. [9]M. Caccamo, L. Zhang, L. Sha, and G. Buttazzo, "An implicit prioritized access protocol for wireless sensor networks," in Proc. IEEE Real-Time Syst. Symp., 2002, pp. 39–48.
9. [10]T. He, J. Stankovic, C. Lu, and T. Abdelzaher, "SPEED: A stateless protocol for real-time communication in sensor networks," in Proc. IEEE 23rd Int. Conf. Distrib. Comput. Syst., 2003, pp. 46–55.
10. [11]P. T. A. Quang and D.-S. Kim, "Enhancing real-time delivery of gradient routing for industrial wireless sensor networks," IEEE Trans. Ind. Inform., vol. 8, no. 1, pp. 61–68, Feb. 2012.
11. [12] S. Bhatnagar, B. Deb, and B. Nath, "Service differentiation in sensor networks," in Proc. Int. Symp. Wireless Pers. Multimedia Commun., 2001.