Review on Reprogramming Protocols and Security Requirements for Wireless Sensor Networks

Vaishali H. Salvi[1] Prof. Dilip Motwani[2]
[2]Assistant Professor
[1]Department of Information Technology [2]Department of Computer Engineering
[1,2]Vidyalankar Institute of Technology, Mumbai, India.

Abstract— The process of disseminating a new program image or changing the functionality of existing code in wireless sensors in a network is known as wireless reprogramming. For security purpose code change should be authenticated preventing an adversary from putting in malicious code. whereas all existing insecure/secure reprogramming protocols are supported the centralized approach, it's important to support distributed reprogramming during which multiple authorized network users will at the same time and directly reprogram sensor nodes.

Key words: WSN, Reprogramming Protocols

I. INTRODUCTION

A wireless sensor network (WSN) contains of a large number of small-sized battery operated nodes that incorporate sensing, computing, and communication capabilities. The applications of WSN in a variety of fields include environmental observation, military functions and gathering sensing data in inhospitable locations. In most applications sensor networks are deployed once and supposed to work unattended for an extended period of time. Management and maintenance tasks of WSNs are difficult.

In rising WSNs consisting number of nodes, reprogramming them one-by-one needs both physical access to every of them (which isn't feasible every time) and consists a particularly long procedure, preventative any period of time reprogramming excluding human intervention. The ability to feature new practicality or perform code maintenance while not having to physically reach each individual node is already a necessary service.

The solutions are available to remotely program the communication devices within the specified network. Here, programming the nodes over the wireless medium introduces different challenges than challenges faced by traditional network reprogramming. Because of requirement of enhanced reliability the real challenge is imposed while transmitting the code to the node. [1]

These systems are operated for extended time periods are unattended so reprogramming the nodes is needed. Additionally adjustments to the environment after deployment and code maintenance and update are needed (e.g. to improve security or robustness). Remote management of such systems is also required to meet the application needs which are changing with respect to time and space.

Many suggested protocols are based on centralized approach where a base station has the authority to reprogram sensor nodes, as shown in the Fig. 1. Unfortunately, the centralized approach is not reliable as a result of, when the base station fails or when any sensor node lose connections to the base station, it is not possible to perform reprogramming. Moreover, there may exist networks having no base station at all, and hence, the centralized approach is not applicable [2]. Also, the centralized approach is inefficient, weakly scalable, and vulnerable to some potential attacks along the long communication path.

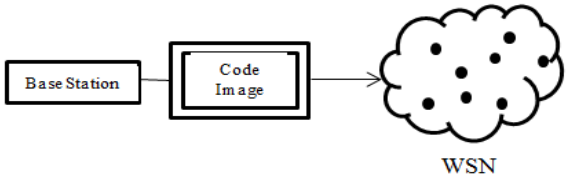

Fig. 1: Centralized Reprogramming Approach

On the other side as shown in the Fig.2 on next page, distributed reprogramming approach permits multiple approved network users to at the same time and directly update program code on completely different nodes while not involving the base station. Also using this approach different approved user can be also assigned for different privileges to reprogram nodes.

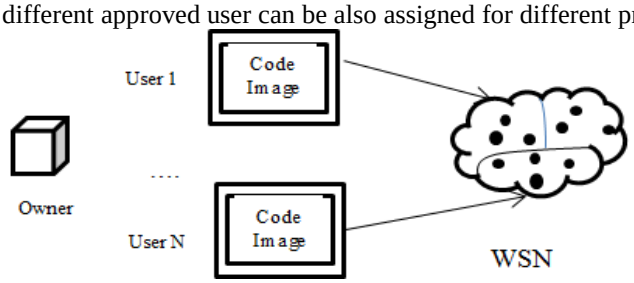

Fig. 2: Distributed Reprogramming Approach

II. REQUIREMENTS AND PROPERTIES OF REPROGRAMMING PROTOCOL

The general process of the network reprogramming is divided in three steps, as shown in Fig 2. The first step prepares the data to be disseminated. In the second step whole dissemination process is carry out. In final step the operating system reconfiguration mechanism translates the received data and uses it to update the program memory.

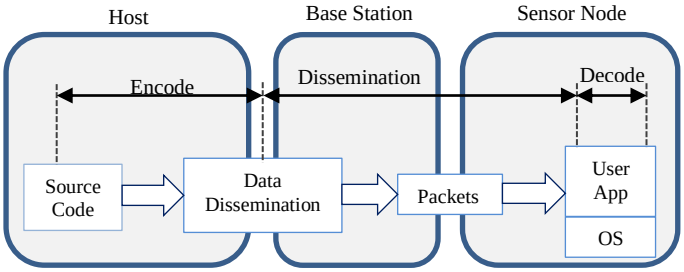

Fig. 3: Network Reprogramming Process

Data dissemination protocols are used to transmit data over the network using its own nodes. The purpose of these protocols is to deliver all data to all nodes accurately. During designing a dissemination protocol we should ensure the fulfillment of the critical properties like low latency, low memory consumption, reliability, energy efficiency, tolerance to nodes insertion/removal, and uniformity [3].

The distributed reprogramming protocol should satisfy the following requirements and properties [1].

A. Authenticity and integrity of code images:

The sender of a code image must be verified by a sensor node before installation, this will ensure that only a trusted source can install a program. Additionally, integrity means that an updated program image cannot be modified undetectably.

B. Freshness:

An older version of a code image cannot be installed over the program with the same or greater version number, ensuring that a node always installs the newest version of a code image.

C. Node compromise tolerance:

A node which is already compromised must be prohibited from causing an uncompromised node to breach the aforementioned security requirements.

D. Distributed:

The authorized network users are able to update the code simultaneously and directly on the nodes without involving the base station and at the same time the protocol should prevent unauthorized users from updating sensor nodes.

E. Supporting different user privileges:

The network owner should designate or limit the level of privilege to each user for ensuring the smooth functioning of WSN. For example, a user is only allowed to reprogram the sensor nodes set with specified identities or/and within a particular localized area for the duration of his subscription.

F. Partial reprogram capability:

In order to prevent total control of sensor nodes by network users overwriting of special modules (for example, authentication module of each new program image) is provided to Network owner only.

G. User traceability:

In most application scenarios, traceability is highly anticipated, particularly for reprogramming.

H. Being efficient:

In general sensor nodes have limited resources (e.g., CPU processing power, memory, bandwidth, and energy). Therefore, the priority should be given to energy efficiency (with respect to both communication and computation) and small storage overhead to cope with the resource-constrained nature of WSNs.

I. Scalability:

The protocol should be efficient for even a large-scale WSN containing thousands of sensor nodes, and also, the large number of users should be supported by the protocol.

III. SECURITY REQUIREMENTS IN WSN

The primary security goal for WSN is to provide confidentiality, integrity, authenticity and availability of information in resource constraints environment. Also, sensor networks can operate in an ad-hoc manner therefore the security goals must cover the goals for both traditional networks requirements and the unique requirements suited to the constraints of wireless sensor networks. Generally, these security goals are classified as primary and secondary. The standard security goals such as Confidentiality, Integrity, Authentication and Availability are primary goals. The secondary goals are Data Freshness, Self-Organization and forward and backward secrecy. The security goals are discussed in detail below [4].

A. Data Confidentiality

Data Confidentiality guarantees that a given message cannot be understood by anyone other than the intended recipients. This is one of the important goals in the network security. A sensor node should not disclose its data to the neighbors. Various encryption schemes can be used to achieve confidentiality.

B. Data Integrity

Data integrity is to make sure that data is not modified in transit either because of malicious intent or accidentally. Though the network has confidentiality measure procedures, there is still a chance that the information integrity has been compromised. Data integrity will be maintained by using various integrity constraints and Message Authentication Code (MAC).

C. Data Authentication

Authentication is required to restrict the activities from unauthorized nodes. It ensures the reliability of the message by recognizing its origin. The recognition of origins is done by validating the uniqueness of the senders and receivers. Data authentication is ensured through cryptographic mechanisms where sending and receiving nodes share secret keys. As a result of the wireless and therefore the unguarded nature of sensor networks, it is very difficult to achieve authentication [5].

D. Data Availability

Data availability determines whether or not a node has the ability to use the resources and whether or not the network is accessible for the nodes to communicate. However, if the base station fails or cluster leader's accessibility is lost, the complete sensor network can be threatened. The necessity of proper security mechanism not only affects the operation of the network but also extremely vital in maintaining the availability of the fully operational network.

E. Data Freshness

Data freshness indicates that the received messages at the node are recent, and previous messages are not being replayed. Importance of data freshness becomes obvious in networks by using shared key operations. To achieve data freshness a randomly generated number, called as nonce or a time dependent counter can be appended to the data. Messages with previous number and/or old time counter are rejected [6]. In this way freshness of data is achieved as this method guarantees acceptance of only recent data.

F. Self-Organization

In typical WSN hundreds of nodes can perform different operations which are installed at various locations. Sensor networks are also ad-hoc networks which have flexibility and extensibility. These are also attractive properties of WSN that pose a serious threat to the overall security situation of the network, which raises the importance of a self-organized and robust structure of network. The lack of self-organization in a sensor network can result into the damage coming from an attack or even the risky as environment may be harmful for the network.

G. Time Synchronization

Most sensor network applications count on some form of time synchronization. An individual sensor's radio may be turned off for periods of time for power conservation. Additionally, the measurement of end-to-end delay of a packet as travelling between two pair-wise sensors is also done by sensors. A more collaborative sensor network may require group synchronization for tracking applications.

H. Secure Localization

Generally, the utility of a sensor network can based on its ability to accurately and automatically locates every sensor within the network. A sensor network premeditated to seek out faults as a result of it required exact location information so as to locate the fault. Unfortunately, an attacker will simply alter non-secured location information by sending false signal strengths and reiterating signals.

IV. REPROGRAMMING PROTOCOLS

A. Deluge

Deluge [7] is a data dissemination protocol and algorithm for broadcasting large amounts of data through a WSN using incremental upgrades for improved performance. It is principally designed at disseminating software image updates, identified by incremental version numbers. The image is disseminated to all nodes in the network. The program image is divided into fixed size pages. Each page is fragmented into fixed size packets so suit the PDU size of the Tiny OS network pile. A bit vector of pages received can also fit in a single packet. Nodes broadcast advertisements containing a version number with a bit vector for any new pages acknowledged, using a variable period based on updating activity. To upgrade part of its image so that it can match with a newer version, a node waits and listens to further advertisements, and then requests the page number or packets required from a selected neighbour. A sender collects a number of requests before picking a page and broadcasting the requested packets. When a node receives the last packet essential to complete a page, it broadcasts advertisement before requesting further pages. This enhances pipelining of the update within the specified network. The state data takes a fixed size, irrespective of the number of neighbors. There are no ACKs or NACKs as the requester determines the date by requesting either packets for a new page, or missing packets from a previous page. Radio network conflict is reduced through heuristics used to select more distant senders. Deluge has support for reducing energy depletion, but is not an energy-aware protocol. It supports dissemination and

activation, but has no specific provision for fault detection and recovery (apart from the watchdog timer under TinyOS); it also has no any feedback mechanisms.

B. Multihop Over-the-Air Programming (MOAP)

MOAP [8] is a multi-hop and over-the-air code distribution mechanism aimed at MICA2 motes running TinyOS[10]. It practices store-and-forward which offers a ripple pattern of updates. A receiver identifies lost segments by using a sliding window, and can re-request them using a unicast message to avoid duplication. A keep-alive timer is used to recover from unanswered unicast retransmission requests. The base-station broadcasts publish messages advertising the version number of the new code. Receiving nodes check received version number against their own version number, and can request to apprise with subscribe messages. A link-statistics mechanism is used so that unreliable links can be avoided. After waiting for some time to receive all subscriptions, the sender then starts the data transfer. Missing segments are requested directly from the sender, which arranges these over further data transmissions. After receiving entire image a node becomes sender itself. If no subscribe messages are received, it transfers the new image to program memory and reboots with the new code. MOAP offers dissemination and activation of updates, with performance optimization to some extent, but no additional major features are provided to support autonomous updates.

C. Multihop Network Reprogramming (MNP)

MNP [9] is aimed at MICA2 motes running TinyOS and the XNP boot-loader. This protocol operates in four phases. During Request mode sources advertise the new version of the code, and concerned nodes make requests. Sources overhear all other advertisements and requests which is a suppression scheme so that network overload can be avoided. During Forward/Download a source communicates a 'Start Download' message to inform the receivers, and then sends the program code a packet at a time. Here is no ACK is used. During second phase i.e. Query/Update the source broadcasts a Query to all its receivers, which retort by unicast asking for the missing packets. After receiving full image, now receiver become source nodes and start advertising. At the time of Reboot, entered when a source receives no requests as a response to an advertisement, the new program image is transmitted to program memory, and the node reboots with the new code. Requests for download are sent to all sources to decrease the hidden terminal effect, and select only single active sender in a neighborhood. MNP supports dissemination and activation of updates, but no additional important features supporting autonomous updates.

D. SDRP

The Secure and Distributed Reprogramming Protocol named SDRP [1], is enlargement of Deluge to be a secure protocol. The idea behind SDRP is to map the identity and reprogramming privilege of a certified user into a public/private key pair. User identity and his reprogramming privilege will be verified using public key, and user traceability and different levels of user authorities are supported. A unique identity-base signature scheme is recommended for distributed reprogramming in WSNs. The projected theme will be considerably reduced efforts on certificate management and therefore the transmission overhead. Since a unique identity-based signature scheme is used in generating the public/private key combine of every approved user, SDRP is efficient for resource -limited sensor nodes and mobile devices in terms of communication and storage needs. The SDRP consists of 3 phases: system initialization, user preprocessing, and sensor node verification. During the system initialization phase, the network owner creates its public and private keys and so assigns the reprogramming privilege and also the corresponding private key to the authorized user(s). Only the public parameters are loaded on every sensor node before deployment. In the user preprocessing phase, if a network user enters the WSN with a new code/program image, it will need to build the reprogramming packets and so send them to the sensing element nodes. In the sensor node verification phase, if the packet verification passes, then the nodes accept the code/program image.

V. CONCLUSION

The organization of sensor node in an isolated environment creates different security loop holes in the network. But the wide application field of WSN encourages and motivates the researchers to make network more and more secure. In this paper various properties of reprogramming properties and reprogramming protocols are discussed. Also, this paper presents security requirements of WSN. Among the protocols discussed above, SDRP is the only distributed protocol which supports multiple users simultaneously and also it is important in large-scale sensor networks used by different users from both public and private sectors. This paper will definitely create interest among the students and future researchers to dream about a reliable, robust and safer wireless sensor network.

REFERENCES

[1] Leligou, H. C., Massouros, C., Tsampasis, E., Zahariadis, T., Bargiotas, D., Papadopoulos, K., & Voliotis, S. Reprogramming wireless sensor nodes. International Journal of Computer Trends and Technology, May to June issue 2011, 7.

[2] Daojing He, Chun Chen, Shammy Chan, Jiajun Bu,"SDRP: A Secure and Distributed Reprogramming Protocol for Wireless Sensor Networks", IEEE, Vol.59, No.11, November 2012.

[3] Steiner, R.; Gracioli, G.; de Cassia Cazu Soldi, R.; Frohlich, A.A. An Operating System Runtime Reprogramming Infrastructure for WSN. In Proceedings of the IEEE Symposium on Computers and Communications, Cappadocia, Turkey, 1–4 July 2012; pp. 621–624.

[4] John Paul Walters, Zhengqiang Liang, Weisong Shi, Vipin Chaudhary, "Wireless Sensor Network Security: A Survey", Security in Distributed, Grid and Pervasive Computing Yang Xiao (Eds), Page3-5, 10-15, year 2006.

[5] Alajmi, Naser. "Wireless Sensor Networks Attacks and Solutions." arXiv preprint arXiv: 1407.6290, 2014.
[6] M. Yasir, "An Outline of Security in Wireless Sensor Networks Threats, Countermeasures and Implementations". Wireless Sensor Networks and Energy Efficiency: Protocols, Routing and Management Book, pp. 507-527, 2012.
[7] Prabal K. Dutta, JonathanW. Hui, David C. Chu, and David E. Culler. (2006). Securing the deluge network programming system. In the 5th ACMIPSN'06, pages 326–333, Nashville, Tennessee, USA. ACM
[8] Stathopoulos, T.; Heidemann, J.; Estrin, D. A Remote Code Update Mechanism for Wireless Sensor Networks; Technical Report CENS #30 for the Center for Embedded Network Sensing; UCLA: Los Angeles, CA, USA, 2003.
[9] Kulkarni, S.S.; Wang, L. MNP: Multihop Network Reprogramming Service for Sensor Networks. In Proceedings of the 25th IEEE International Conference on Distributed Computing Systems, Baltimore, MD, USA, 14–16 June 2005; pp. 7–16.
[10] Tiny OS, "An open-source OS for the networked sensor regime," 2012. [Online] Available: http://www.tinyos.net/