Effective Selective Encryption Method for WSN Using One-way Hash Chain Algorithm

S. Saranya[1] M.Sc., Part –time M.Phil Scholar, Department of Computer Science, Navarasam Arts & Science College for Women, Arachalur.

Mrs. M.Sumathi[2] M.Sc., M.Phil., Asst.Professor, Department of Computer Application, Navarasam Arts & Science College for Women, Arachalur.

Abstract - **Sensing devices are widely used to build and deploy self-organizing wireless sensor networks for a variety of critical applications such as smart cities, smart health, precision agriculture and industrial control systems. A Data Stream Manager (DSM) at the server collects the data streams (often called big data) to perform real time analysis and decision-making for these critical applications. A malicious adversary may access or tamper with the data in transit. One of the challenging tasks in such applications is to assure the trustworthiness of the collected data so that any decisions are made on the processing of correct data providing multilevel data confidentiality along with data integrity for big sensing data streams in the context of near real time analytics is a challenging problem. In this paper, we propose a Selective Encryption (SEEN) method to secure big sensing data streams that satisfies the desired multiple levels of confidentiality and data integrity. Our method is based on two key concepts: common shared keys that are initialized and updated by DSM without requiring retransmission and a seamless key refreshment process without interrupting the data stream encryption/decryption. In this project, a novel scheme to secure a multihop network programming protocol through the use of multiple one-way hash chains is proposed. The scheme is shown to be lower in computational, power consumption and communication costs yet still able to secure multihop propagation of program images.**

Keywords: **Data Stream, Encryption, Multihop, Wireless Sensor, Classification, Hash chain model.**

1. INTRODUCTION

Network programming is becoming necessary for wireless sensor networks (WSNs) because there is always a need to fix bugs in program images or insert new functionalities after WSN is deployed in an evolving, dynamic environment.

Early network programming protocols concentrated on reliable program image dissemination and minimal end-to-end update latency using distribution methods having epidemic-like characteristics. However, they provided no authentication or security mechanisms. The absence of authentication of the broadcast of program image means that a malicious node could install arbitrary program images in the sensor nodes. An adversary could just capture one sensor node, inject malicious program images into the network, and thereby take control of the entire WSN.

The goal of this paper is to present a design and implementation for a new scheme to verify the authenticity and integrity of program updates in network programming protocols. The work is motivated by the following challenges. First, as a significant class of WSN sensor nodes is resource-impoverished, traditional cryptographic schemes are impractical.

For example, the Tmote has 10 KB RAM, 48 KB flash memory, 1 MB storage, and 250 kbps communication bandwidth. This is barely sufficient to execute traditional asymmetric cryptography (e.g., RSA or Diffie and Hellman). It is important that a security scheme for WSNs should be low in power consumption and have low computational overhead.

The second challenge arises from the open wireless environment in which WSNs are typically deployed. Since program updates are also broadcast through the wireless medium, an adversary can readily intercept the program updates and attempt to forge a malicious program image while avoiding detection. Another complication arises from the way that sensor nodes are deployed.

Typically, these are left unattended after deployment, and as such are at risk of physical or functional capture. It is possible to physically secure a sensor network node against theft or tampering by a variety of means, but these physical approaches areoutside the scope of this paper. The thesis presents a scheme that is resilient against brute-force attack and node compromise. The main contribution of the paper is,

- To introduce a novel scheme to secure a multihop network programming protocol through the use of multiple one-way hash chains.
- To apply the scheme such that it should be lower in computational, power consumption, and communication costs yet still able to secure multihop propagation of program images.
- To demonstrate the use of this scheme and provide some results using a network programming protocol.
- To evaluate the performance of the scheme
- To provide a cost-effective security scheme for network programming;In the proposed scheme, a digital signature is not required to bootstrap the network programming process. Instead, multiple distinct one-way key chains are distributed to the nodes based on

their hop distances to base station for packet authentication. This obviates the need for digital signature, resulting in efficient use of the limited computational power of each sensor node.

- To apply immediate verification; in the proposed scheme, no time synchronization among sensor nodes is required, and consequently, the packets of a program update can be verified immediately as they are received.

II. LITERATURE SURVEY

Thanos Stathopoulos, John Heidemann and Deborah Estrin stated that wireless sensor networks consist of collections of small, low-power nodes that interface or interact with the physical environment. The ability to add new functionality or perform software maintenance without having to physically reach each individual node is already an essential service, even at the limited scale at which current sensor networks are deployed. TinyOS supports single-hop over-the-air reprogramming today, but the need to reprogram sensors in a multihop network will become particularly critical as sensor networks mature and move toward larger deployment sizesUsing the Rsync algorithm, the host program generates the difference.

- The Rsync algorithm calculates a checksum pair (check-sum, hash) for each fixed sized block (e.g. B bytes) of the previous program image and the checksum pair isinserted into a hash table for look-up.
- Rsync reads the current program image and calculates the checksum for the B byte block at each byte. If it finds a matching checksum in the hash table, Rsync calculates the hash for the block and compares it with the corresponding entry in the hash table. If the hash also matches, then the block is considered a matching block.
- Rsync moves to the next byte for comparison if the block doesn't have a matching checksum or a hash. A region of bytes that doesn't have any matching blocks is tagged as non-matching block and needs to be sent explicitly for rebuild.

Suppose the program image is shifted in the middle due to a modification operation. Rsync forms a B byte window and calculates the hash for it. If the modified bytes are different from any blocks in the previous program image, then the hash of the modified bytes doesn't match any hash table entry with very high probability. Rsync moves the window one byte at a time and calculates the checksum for any possible match. It doesn't match until Rsync starts to read unmodified blocks. At this moment, Rsync has found a matching block.

Delugeis a multi-hop network programming mechanism and it has two main functions: dissemination of block data and reprogramming boot loader. Deluge disseminates the program code in an epidemic fashion to propagate the program code while regulating the excess traffic. In order to increase thetransmission throughput, Deluge uses optimization techniqueslike adjusting packet transmission rate and spatial multiplexing. Unlike MOAP, Deluge uses a fixed sized page as a unit of buffer management and transmission.

Reijers et al.developed an algorithm that can efficiently encode the program code update. To describe the difference between the two program codes, the host program generates "edit script" which consists of operations like copy, insert, address repair and address patch. These operators modify the program code at the instruction level. Using complex operations like address repair and address patch helps reduce the network traffic, but it increases EEPROM accesses.

Kapur et al.implemented an incremental network programming based on Reijets et al's encoding protocol and MOAP multi-hop network programming protocol. Their implementation supports the incremental multi-hop delivery of native code, but it has a limitation similar to Reijers et al's algorithm. It is dependent on specific processor instruction set and does not provide a general solution. The implementations above transmit the program code in native code. Whereas Mat´ e transmits the virtual machine code which is an application specific code for Mat´ e virtual machine.
One advantage is that a sender node doesn't need to send the network programming module because it is assumed that the virtual machine is already running on the receiver node. This allows Mat´ e to distribute program code quickly. One drawback of Mat´ e is that it executes only the virtual machine instructions and a regular sensor application should be converted to the virtual machine instructions before execution.

Trickle is an improvement over Mat´ e. In Mat´ e, eachsensor node distributes the code by flooding the network with packets. This can lead to the network congestion and the algorithm cannot be used for a large sensor network. Trickle uses an epidemic algorithm to propagate the program code only to the sensor nodes that needs to be modified. Their implementation supports incremental delivery of the native code. Unlike previous approaches, they generated the program code difference by comparing the program code in block level without any prior knowledge of the program code structure. This gives a general solution that can be applied to any hardware platform.

III.METHODOLOGY

The section describes the design of the secure network programming scheme. Deluge has a hierarchical organization of program images (program images → pages → packets). The scheme works at the packet level and consists of two phases: first, initialization and key pre-distribution, and second, packet preprocessing and verification.

Resistance to node compromise*:* Our scheme is resilient tonode compromise regardless of the number ofcompromised nodes, as long as the subset of noncompromisednodes can still form a connected graph.

Low computational overhead*:*The proposed scheme employssymmetric cryptography to authenticate broadcastprogram updates. As a result, the computationaloverhead is relatively low compared to the existingwork in secure network programming.

Low communication overhead*:*Theproposed scheme, thecommunication overhead depends on the numberof hops needed to reach all nodes in the network.

Robustness to packet loss*:*The proposed scheme inherits robustnessto packet loss from underlying protocolcomponents. For example, the system itself providesreliable data dissemination.

Immediate authentication*:* Time synchronization betweenbase station and sensor nodes is not requiredin the

scheme. This allows immediate authenticationof packets as they are received.

The main methodology as follows,

- **Data set collection**

The length of the Hash Chain (In computer security, a **hash chain** is a method to produce many one-time keys from a single key or password) is denoted by L. The Hop group count is denoted by S. The L and S value are keyed in to the database table 'Parameters'. Text box controls are placed to key in the L and S values. The committed values i.e., final key in the hash chains are up to S count. Each Hop group nodes are given with a committed value so that they can decrypt the data. The data to be sent from base station to all nodes will split into L packets and so the hash chain will have L+1 key.

- **Network configuration**

The Hop index is selected and the Node Id is keyed in to the database table 'Nodes'. The node id for first group is denoted as A1, A2, A3, etc, The node id for second group is denoted as B1, B2, B3, etc, and goes on up to 'S' number of Hops. All the nodes in various hop groups need to transmit data after processing packet information and cut out data of their corresponding part in the packet. For that, the nodes use the committed values already pre-distributed by the base station.

- **Selective encryption model**

The hash chain is built with giving a prime number in $K_{L,1}$level and creating keys of upto 'L' count. The hash chain is built for 'S' count. The committed value (i.e., the last key) produced are given to all nodes. The key details are stored in 'KeyValues' table.During packet preprocessing, the key values are fetched from 'KeyValues' and Packet Data is concatenated with key values and are broadcasted to all the nodes.

- **One Way Hash Chain Model**

During packet receiving, the raw packet data is extracted from the received packet based on the given algorithm. The node receives the data and cut out the data of their corresponding part in the packet. For that, the nodes use the committed values already pre-distributed by the base station. Only the nodes already becomes the part of the network and having the hash chain's committed value can process the data. The node without having committed value, if tries to process the packet, then the packet data cannot be parsed out by the node and the network treats that nodes as suspicious, i.e., attacker node.

- **Hybrid hash chain model**

The parameter setting for 'S' Hop group count and 'L' number of packets or number of key values in hash chain is displayed using this report. The data grid view is filled with a data table which contains 'parameter' table's record. During packet preprocessing, these values are retrieved out and used in the algorithms.

IV. CONCLUSION

An authentication scheme is proposed to secure multihop network programming with multiple one-way hash chains. Instead of the expensive asymmetric cryptographic primitives used in much prior work, the scheme employs only symmetric cryptographic primitives, in a circular geographic node deployment model. Possible attacks an adversary could mount on the scheme is discussed and provided simple and effective counter measures against them. Finally, it provides a comprehensive performance evaluation of the scheme in terms of end-to-end latency and power consumption, which is said to be believed, is the first power consumption evaluation of a security scheme for network programming protocols. The application works well for given tasks in network environment. Any node with .Net framework installed can execute the application. The underlying mechanism can be extended to any or all kind of platform like Linux, Solaris and more. The system eliminates the difficulties in the existing system. It is developed in a user-friendly manner. The system is very fast and any transaction can be viewed or retaken at any level. Error messages are given at each level of input of individual stages. This software is very particular in securing the multi-hop network programming.

V. SCOPE FOR FUTURE ENHANCEMENTS

The future work will include the design of security models for network programming. The application if developed platform independent it can be used in may operating systems. Most practical wireless networks will be restricted to fewer hops than four since the throughput significantly degrades for larger hop counts. In addition, large deployments will be likely to form some kind of hierarchy. Hence, the utilization of memory can be measured in future.Currently the scheme has a slightly less memoryoverhead, while in the more complex applications; the scheme may utilize more memory. The future study can be in the area of more significant memory savings. The new system becomes useful if the above enhancements are made in future. The new system is designed such that those enhancements can be integrated with current modules easily with less integration work.

REFERENCES

[1] Arasu, et al. "STREAM: the stanford stream data manager (demonstration description)." In *ACM SIGMOD international conference on Management of data*, pp. 665-665, ACM, 2003.

[2] H-S. Lim, Y-S. Moon and E. Bertino, "Provenance-based trustworthiness assessment in sensor networks." In SeventhInternational Workshop on Data Management for Sensor Networks, pp. 2-7. ACM, 2010.

[3] S. Sultana, G. Ghinita, E. Bertino and M. Shehab, "A lightweight secure provenance scheme for wireless sensor networks." In *18th International Conference on Parallel and Distributed Systems (ICPADS),* pp. 101-108, 2012.

[4] [4] R. A. Shaikh, S. Lee, M. AU Khan and Y. J. Song, "LSec: lightweight security protocol for distributed wireless sensor network." In *IFIP International Conference on Personal Wireless Communications*, pp. 367-377. Springer Berlin Heidelberg, 2006.

[5] G. Selimis et al., "A lightweight security scheme for wireless
body area networks: design, energy evaluation and proposed microprocessor design." *Journal of medical systems*, vol. 35, no. 5, pp. 1289-1298, 2011.

[6] G. Selimis, at al. "Evaluation of 90 nm 6 T-SRAM as Physical Unclonable Function for Secure Key

Generation in Wireless Sensor Nodes", *in IEEE ISCAS Brazil*, pp. 567-570, 2011.

[7] M. Roesch,"Snort: Lightweight Intrusion Detection for Networks." *LISA*, vol. 99, no. 1, pp. 229-238. 1999.

[8] N. Tsikoudis, A. Papadogiannakis and E. P. Markatos, "LEoNIDS: a Low-latency and Energy-efficient Network-level Intrusion Detection System." *IEEE Transactions on Emerging Topics in Computing*, vol. 4, no. 1, pp.142-155, 2016.

[9] W. Lee and S. J. Stolfo, "Data Mining Approaches for Intrusion
Detection." In *Usenix security*. 1998.

[10] Y. Xie, D. Feng, Z. Tan and J. Zhou, "Unifying intrusion
detection and forensic analysis via provenance awareness." *Future Generation Computer Systems*, vol. 61, pp.26- 36, 2016.