Securing Wireless Sensor Network (WSN) Using Embedded Intrusion Detection Systems

Qutaiba I. Ali* Sahar Lazim Enaam Fathi
Computer Engineering Department/College of Engineering/Mosul University/Iraq
***E mail: Qutaibaali@uomosul.edu.iq**

Abstract

This paper focuses on designing distributed wireless sensor network gateways armed with Intrusion Detection System (IDS). The main contribution of this work is the attempt to insert IDS functionality into the gateway node (UBICOM IP2022 network processor chip) itself. This was achieved by building a light weight signature based IDS based on the famous open source SNORT IDS. Regarding gateway nodes, as they have limited processing and energy constrains, the addition of further tasks (the IDS program) may affects seriously on its performance, so that, the current design takes these constrains into consideration as a priority and use a special protocol to achieve this goal. In order to optimize the performance of the gateway nodes, some of the preprocessing tasks were offloaded from the gateway nodes to a suggested classification and processing server and a new searching algorithm was suggested. Different measures were taken to validate the design procedure and a detailed simulation model was built to discover the behavior of the system in different environments.

Keywords: Wireless Sensor, Network security, Intrusion Detection System, Gateway, Network Processor

تامين شبكة المتحسّس اللاسلكية (WSN) باستخدام أنظمة كشف التسلل المطمورة

قتيبة ابراهيم علي سحر لازم قدوري انعام فتحي خضر
هندسة الحاسبات/ كلية الهندسة/ جامعة الموصل/ العراق

الخلاصة

يركز هذا البحث على تصميم بوابات شبكة المتحسّسات اللاسلكية الموزعة المزودة بنظام كشف التسلل (IDS). ان المساهمة الرئيسية في هذا العمل هو محاولة لادخال وظيفة نظام كشف التسلل إلى بوابة الشبكة (IP2022 UBICOM رقاقة معالج الشبكة) نفسها. وقد تحقق ذلك من خلال بناء لنظام كشف التسلل استنادا إلى احد اشهر برامج كشف التسلل المفتوح المصدر وهو ال SNORT. فيما يتعلق بخصائص بوابة الشبكة من محدودية المعالجة والطاقة ان إضافة المزيد من المهام (برنامج IDS) قد يؤثر بشكل خطير على أدائها، لذلك فان التصميم الحالي يأخذ هذه القيود بعين الاعتبار بوصفها أولوية وتم استخدام بروتوكول خاص لتحقيق هذا الهدف. لأجل تحسين أداء بوابة الشبكة ، بعض مهام المعالجة الأولية كانت قد حملت مسبقا من بوابة الشبكة إلى خادم التصنيف والمعالجة المقترح ، كما تم اقتراح خوارزمية بحث جديدة. واتخذت تدابير مختلفة للتحقق من صحة التصميم الداخلي وتم بناء نموذج محاكاة مفصل لاكتشاف سلوك النظام في بيئات مختلفة.

1. Introduction

A wireless sensor network (WSN) is a wireless network consisting of spatially distributed autonomous devices using sensors to monitor physical or environmental conditions. A WSN system incorporates a ***gateway*** that provides wireless connectivity back to the wired world and distributed nodes. The selected wireless protocol depends on the application requirements. Some of the available standards include 2.4 GHz radios based on either IEEE 802.15.4 or IEEE 802.11 (Wi-Fi) standards or proprietary radios, which are usually 900 MHz [1].

There were many attacks that have been identified in WSN. These attacks can be classified on various criteria, such as the domain of the attackers, or the techniques used in attacks. It can be roughly classified as: passive or active, internal or external, attacks on protocol layer, stealthy or non-stealthy, cryptography or non-cryptography related. Attacks on protocol layer are often referred to as Denial-of-Service (DoS) attacks which target any layer of a sensor network [2-5].

Most research on security in sensor networks has focused on prevention techniques, such as secure routing protocols, cryptography and authentication techniques. These security mechanisms are usually the first line of defense. However, experience with the Internet has shown that flaws in these protocols are continuously being found and exploited by attackers. Consequently, it cannot be relied on intrusion prevention techniques alone [2][6]. In practice, Intrusion Detection Systems (IDSs) are needed to detect both known security exploits and even novel attacks that have yet to be experienced [7-9]. In this paper, a wireless intrusion detection system is adapted from the Snort IDS engine, which it is focused on upper three layers and it is located in the gateway nodes between the server and other sensor nodes. The remainder of this paper is organized as follows: section 2 includes the related works. Section 3 demonstrates the protocol actions of the proposed IDS in wireless sensor network. Section 4 explains the conversion algorithm that implemented on conversion server to make the sensor node in update case against new attacks. Section 5 includes the hardware implementation of WSN gateway armed with IDS on UBICOM platform and implementation of searching algorithm on the proposed WSN gateway with the results. Section 6 contains a simulation of the proposed system and the simulation results with a discussion and finally, section 7 provides conclusions and future works.

2. Related Works

There were many papers that consider some of the most significant WSN security problems. In [2], the authors presented a method for intrusion detection in wireless sensor networks. The proposed intrusion detection scheme uses a clustering algorithm to build a model of normal traffic behavior, and used the model of normal traffic to detect abnormal traffic patterns. A key advantage of the proposed approach is that it is able to detect attacks that have not previously been seen. In [10], the authors proposed a Battery-Sensing Intrusion Protection System (B-SIPS) for mobile computers, which alerts on power changes detected on small wireless devices, using an innovative Dynamic Threshold Calculation algorithm. The master thesis [3] introduced the problem of Intrusion Detection in sensor networks and took a step towards the development and implementation of a lightweight detection architecture that can be applied to such networks. In [8], the authors proposed a modular, scalable, secure and trusted networking protocol stack, able to offer self-configuration and secure roaming of data and services over multiple administrative domains and across insecure infrastructures of heterogeneous WSNs. In [1], the authors described the security threats and various kinds of attacks on WSNs and described the need for an intrusion detection system (IDS) in WSNs. In [11], the authors proposed an approach of an Intrusion Detection Based Security for Cluster-Based Wireless Sensor Networks. In the proposed methodology, an efficient MAC address based intruder tracking system has been developed for early intruder detection and its prevention.

3. Details of the Suggested System

The main contribution in this paper is the attempt to insert IDS functionality into the gateway node itself. This was achieved by building a light weight signature based IDS based on the famous open source SNORT IDS. *SNORT* is an open source network IDS capable of performing real-time traffic analysis and packet logging on Internet Protocol (IP) networks. *SNORT* can perform protocol analysis and content searching/matching, and can be used to detect a variety of attacks and probes. *SNORT* uses a simple, lightweight rules description language that is flexible and quite powerful. Snort rules are divided into two sections, the rule header and the rule options. The header contains the rule's action, protocol, source and destination IP addresses including network masks, and the source and destination ports. All options are defined by keywords specifying which fields of the packet should be inspected, such as TTL and content. The content keyword contains a signature or data pattern to be matched against the packet content [12]. Based on the previous discussion, intrusion detection can be divided into two problems: packet filtering (or classification based on header fields) and string matching over the packet payload.

Regarding gateway nodes, as they have limited processing and energy constrains, the addition of further tasks (such as an IDS program) may affect seriously on its performance, so that, the current design takes these constrains into consideration as a priority and the following procedure was followed:

1. The gateway nodes were loaded with specified rules set (not all rules) which represent the most common attacks at that time. The determination of these rules as "important" is achieved using IDS sensors (which are specific PCs with complete and up to date SNORT program installed on them) distributed around the network. These sensors monitor the network status (from security point of view) and prepare a report of the most common attacks at that time. These reports are sent to Classification and Processing Server for further processing.
2. Classification and Processing Server collects the reports from IDS sensors and analyzes them to assign the most common risky attacks at that time. Also, it has the classification and processing program which is used to classify the SNORT rules. After that, the server will broadcast the processed rules set to all wireless gateway nodes (after notifying them) that exist in the network. **Fig. 1** shows the description of the suggested system.

In order to keep the efficiency and performance of the gateway nodes, a new rules processing algorithm is suggested. The main idea of the suggested algorithm can be abstracted through implementing the preprocessing part of algorithm which acts the building tree for packet filtering and building tree for string matching in the processing and classification server and only the searching part of algorithm (which acts the searching tree of packet filtering and searching tree of string matching) is implemented on the gateway node. The preprocessing part or building trees part will be sent as an updated file from processing and classification server to the gateway nodes.

4. Software Implementation Challenges

This section discusses the software details of the suggested classification and searching algorithm. As mentioned earlier, in order to optimize the performance of the gateway nodes which were supplied with the IDS engine, some of the preprocessing tasks were offloaded from the gateway nodes to classification and processing server.

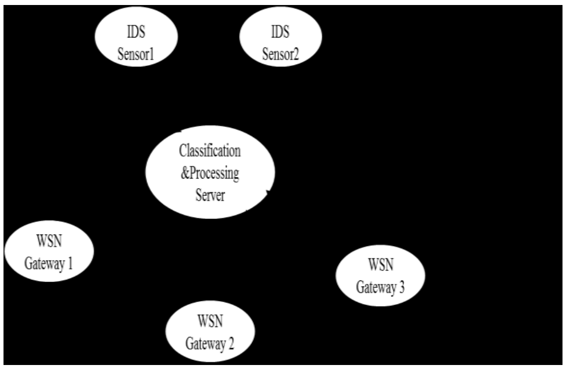


Figure1. Description of the suggested system

The main idea of the suggested classification and searching algorithm is shown in **Fig. 2**. It can be abstracted through implementing the preprocessing part of algorithm which acts the building tree for packet filtering and building tree for string matching (will be explained in next subsection 4.1.1 and 4.2.1) on processing and classification server. Only the searching part of algorithm which acts the searching tree of packet filtering and searching tree of string matching (will be explained in next subsection 4.1.2 and 4.2.2) is implemented on the proposed WSN gateway.

Figure2. The suggested classification and searching algorithm

4.1 Packet Classification

Before the suggested algorithm for packet classification is discussed, the range lookup problem must be solved. Algorithms that perform classification in multiple dimensions often use a one-dimensional lookup algorithm as primitive and field specifications can be arbitrary ranges. The range lookup problem is solved here by first converting each range to a set of maximal prefixes, and then solving the prefix matching problem on the union of the prefixes thus created. Hence, if a given range is split into the minimum number of sub ranges satisfying this property, a set of maximal prefixes resultant will be equivalent to the original range [13].Table 1 lists examples of some range to prefix conversions for 4-bit fields.

Table 1. Examples of range to prefix conversions for 4-bit fields.

Range	Constituent maximal prefixes
[4,7]	01**
[3,8]	0011, 01**, 1000
[1,14]	0001, 001*, 01**, 10**, 110*, 1110

It can be seen that a range on a -bit dimension can be split into a maximum of maximal prefixes. Afterward, the suggested packet classification algorithm has two parts. First part is used to build a tree for all the rules headers fields, but the second part is used to search through the building tree to find the matched rule for the input packet. An example real-life classifier in five dimensions is shown in Table 2. By convention, the first rule R1 is of highest priority and rule R12 is of lowest priority. These parts are discussed in the following subsections.

Table 2. Examples of real-life rule classifier

rule	Srcperfix	Destperfix	Src port	Dest port	protocol
R0	010*	10*	0,65535	25,25	6
R1	101*	001*	53,53	443,443	4
R2	*	10*	53,53	1024,65535	17
R3	*	01*	53,53	443,443	4
R4	1*	1*	53,53	25,25	4
R5	1101*	001*	0,65535	2788,2788	17
R6	110100*	11*	53,53	5632,5632	6
R7	*	11*	53,53	25,25	6
R8	111*	01*	67,67	5632,5632	17
R9	010*	10*	0,65535	0,1023	6
R10	111*	1*	67,67	25,25	4
R11	1101*	001*	53,53	2788,2788	4
R12	111*	*	1024,65535	5632,5632	4

4.1.1 Building Tree

In our implementation, the classifier processes the five main header fields: the IP

Destination Address, IP Source Address, IP Destination Port, IP Source Port, and Layer 4 Protocol Type. Hence, a 5-dimensional classifier is built, which uses a 5-stagehierarchical search trie. In each stage, the classification algorithm processes a single packet header field. It first builds a source prefix trie, and each prefix node of the source trie hierarchically connects a destination prefix trie which comprises of rules with the same source prefix field and so on for the rest header fields until reach to the last header field of the classification algorithm. **Fig. 3** shows the first two stage of the tree where the big trie is the source prefix trie and multiple small tries are destination tries. Dark nodes represent prefix nodes or rule nodes, and white nodes represent empty internal nodes which are not associated with a prefix or a rule. A packet may match more than one rule, thus each rule in the database is associated with a field that identifies its priority.

4.1.2 Searching Tree

During searching phase, a match with a node is determined if it has prefix match with the source and the destination prefixes, exact match with the protocol, and range match with the source port and the destination port. The searching phase in the suggested algorithm is immediately finished without searching the complete trie if an input packet matches a priority rule. This property effectively improves the searching performance.

The searching proceeds to the left or right according to the sequential inspection of destination address bits starting from the most significant bit. If there is a match with all the fields in a tree, it is considered as "match" and its priority number is remembered. The searching will be stopped immediately in case if it ends with a match with a priority rule or at a leaf while it is always finished at a leaf in other trie-based algorithms.

For example as shown in **Fig. 3**, assume a given input packet (110101, 101000, 53, 25, 4), the destination address '101000' is considered. At the root node, the input packet does not match R12 in port numbers. Since the first bit of the input destination address is 1, searching moves to aright child of the root node. The input is compared with R4 and ends up with a match. Hence R4 is remembered as the current best match. The rule R10 which is stored in the same node is not necessarily compared since it has a lower priority than the current best match. Search moves to the left child since the second bit of the destination address is 0. Since the stored rule R0 and R2 have higher priorities than the current best match, the input is compared with them. The input does not match them. The rule R9 is not compared since it has a lower priority. Now the search is over and the best match of the current input is R4.

If more than a rule has the same prefix in that field, the rules are most likely to be stored in the same node if they are not stored in other nodes as a priority rule. Those rules are linearly compared. The linear search in a node deteriorates the search performance.

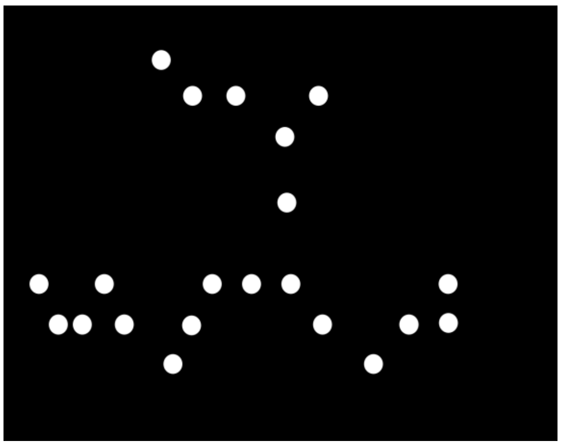


Figure3. Source and destination prefix trie of the tree

4.2 String Matching

String matching or content matching is the main task in Snort program. From a stream of packets, the algorithm identifies those packets that contain data matching the signatures of a known attack. The problem of pattern matching is well researched, many algorithms exist and they can be classified into either single pattern string matching or multiple pattern string matching. In single pattern string matching the packet is searched for a single string at a time. On the other hand, in multiple pattern string matching the algorithm searches the packet for the set of strings all at once. Aho-Corasick (AC) is a multiple pattern string matching algorithm, meaning it matches the input against multiple strings at the same time. Multiple pattern string matching algorithms generally preprocess the set of strings, and then search all of them together over the packet content [12]. Aho-Corasick (AC) algorithm is chosen in the classification and processing algorithm. The following subsections illustrate the building and searching phase for this algorithm.

4.2.1 Building Tree

The essence of the Aho-Corasick algorithm involves a preprocessing step which builds up a state machine that encodes all of the strings to be searched. The state machine is generated in two stages. The first stage builds up a tree of the strings that need to be identified for each node in the tree of previews section that contains one rule or more. The root of the tree represents the state where no strings have been even partially matched.

4.2.2 Searching Tree

Before accepting any input characters and at the beginning of each packet, all tiles are reset to start from idle state. To match a string, the root node and traverse edges start according to the input characters observed. Each node in the previews tree holds the entire set of strings that are to be searched, a character is read from an incoming packet, and computes the set of matches. Matches can be reported either after every byte, or can be accumulated and reported on a per-packet basis. When a string match is not found it is possible for the suffix of one string to match the prefix of another. To handle this case failure edges are inserted which shortcut from a partial match of one string to a partial match of another.

5. Hardware Implementation Challenges

In order to exploit the features of the embedded system to implement the wireless gateway, UBICOM network processor is used. UBICOM IP2022 is a produced by UBICOM Company; it provides the whole solution as a fully integrated platform - the Real Time Operating System (RTOS), the protocol stack, and the necessary hardware. UBICOM's IP2022 chip embeds some basic hardware, but it permits combining it with on-chip software to support the most prevalent protocols. The same device can supports Ethernet, Bluetooth wireless technology, IEEE 802.11, and so on. The key to this approach is Software System on Chip (SOC) technology [14].

5.1 Implementing WSN gateway armed with IDS on UBICOM Platform

The software construction of the proposed WSN gateway armed with IDS deals with processing headers phase and testing against rules phase (see **Fig. 4**). In processing headers phase, the IP and Transport layer headers are processed through reading header's fields successively from UBICOM memory. Moreover, during headers processing, checksum will be calculated for header or for header and data, to warrant no errors during receiving the packet. If any error happens such as: checksum not equal zero, header length is greater than the total length or other errors, the packet is disregarded. After the processing phase is completed, the headers parameters are delivered to the testing phase to check them against IDS rules in order to find attacks.

5.2 Implementation of Searching Algorithm on the Proposed WSN Gateway

As mentioned earlier in section 4, the searching algorithm on the proposed WSN gateway is divided to two parts: packet header filtering or classification and string

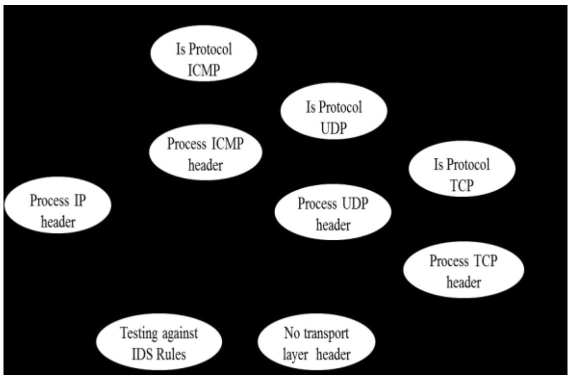


Figure4. State diagram of the suggested WSN for receiving packet

matching and each one has building and searching phases. In order to evaluate the performance of each algorithm separately, several tests were implemented. In this section, measurements of the searching algorithms of each phase were performed as three steps. In the first one, results were discussed by computing the memory storage based on the number of rules that can be stored in the memory of the WSN gateway. While in the second one, results were discussed by calculating the preprocessing time to build the searching tree. Finally, results were discussed by manipulating the searching time to find the specific headers rule or string.

5.2.1 Results for Packet Filtering (Classification Phase)

The worst case is computed considering a classifier table filled with totally different rules, without overlapping in the values of the fields. The packet filtering is based on five headers field (as mentioned in section 4.1), so, it will be searched for these headers fields for each incoming packet.

- *Storage Memory Requirements:* it represents the amount of the on chip memory (SRAM data memory) that is used to store the database of header. Its size depends on the number of rules (N), the number of headers field that are checked for filtering (d), and the width of each header field in bits (w). The relationship among these variables O(Ndw) is shown in **Fig. 5**, it can be noted that the memory storage increases when each variable is increased where, each graph are created with d=5 and various number of rule sets.

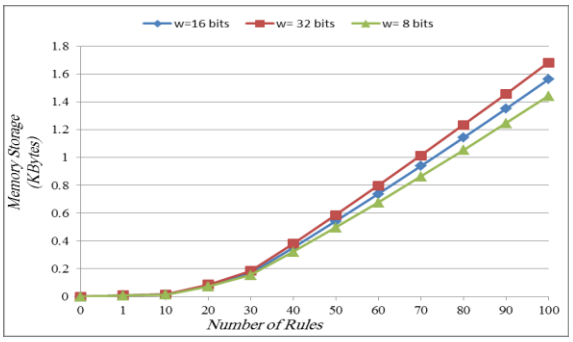


Figure5. Memory storage of set rules for packet filtering phase

- *Preprocessing Time of Packet Filtering:* it represents the amount of time to build the tree through the database of headers fields. It depends on the number of header fields that are checked for filtering (d), and the width of each header field in bits (w). The relationship among these variables is O(d^2w). In **Fig. 6**, it can be noted that processing time increases when each variable is increased where, the graph is created with d=5 and different sizes of headers field width.

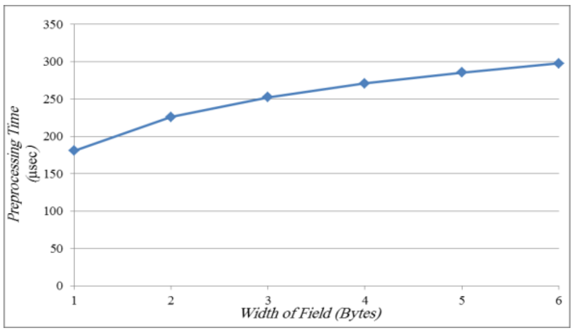


Figure6. Preprocessing time for various width of field

- *Searching Time of Packet Filtering:* it represents the time elapsed to find a specific header fields that matched the header fields for the input packet. It depends on the number of headers field that are checked for filtering (d), and the width of each header field in bits (w). The relationship among these variables is O(w^d). In **Fig. 7**, it can be noted that searching time increases when each variable is increased where, the graph is created with d=5 and different sizes of headers field width.

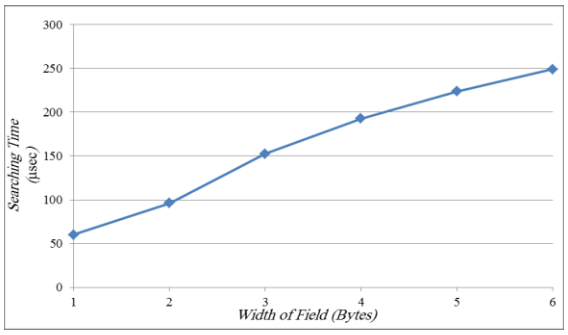


Figure7. Searching time for various width of field

5.2.2 Results for String Matching Phase

String matching performance depends on number of matches between the packet payload and searching arrays; hence, we considered the scenario (as worst case) in which all patterns exists in the packet payload. After successful matching rules is found in the header's packet, the proposed WSN gateway will start performing string matching for the incoming packet.

- *Storage Memory Requirements:* the storage requirement is another important metric to evaluate the string matching algorithms, where it represents the on chip memory that stores the searching arrays. It depends on the number of searching arrays in bytes (n) and length of each searching arrays (L). The relationship among these variables is O(Ln). In **Fig. 8**, it can be noted that storage of memory increases when each variable is increased where, the graph is created with L=15 and various number searching arrays.
- *Preprocessing Time of String Matching:* It represents the time spent to build the tree for a set of searching arrays. It depends on the number of searching arrays in bytes (n) and length of each searching arrays (L). The relationship among these variables is O(log(Ln)). In **Fig. 9**, it can be noted that processing time increases when each variable is increased where, the graph is created with L=15 and various number searching arrays.

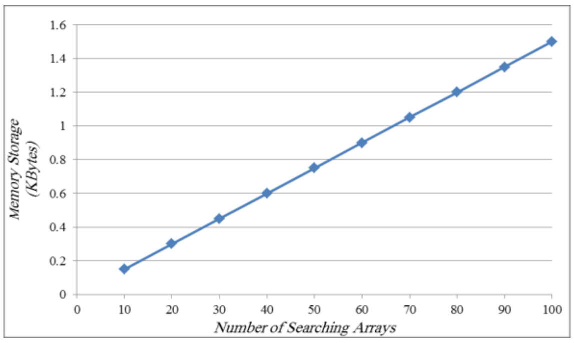


Figure8. Memory storage of searching arrays for string matching phase

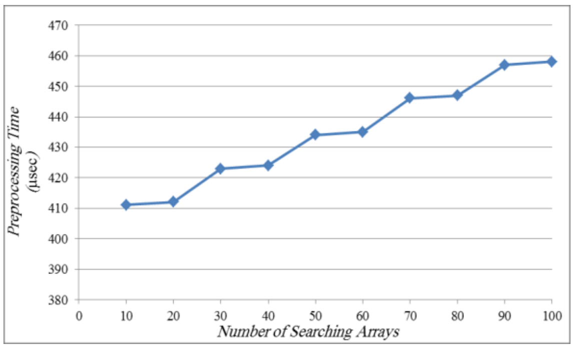


Figure9. The preprocessing time for different sizes of searching arrays

- *Searching Time of String Matching:* it represents the consumed time to find the specific searching array from set of searching arrays that matches the pattern at the packet's payload. The results are generated with packet size of 1300 byte and 100 patterns; the results are conducted with pattern length 15 byte. It depends on the length of payload in bytes (m). The relationship among these variables is O(m+1). In **Fig. 10**, it can be noted that searching time increases when the variable is increased where, the graph is created with L=15 and various number searching arrays.

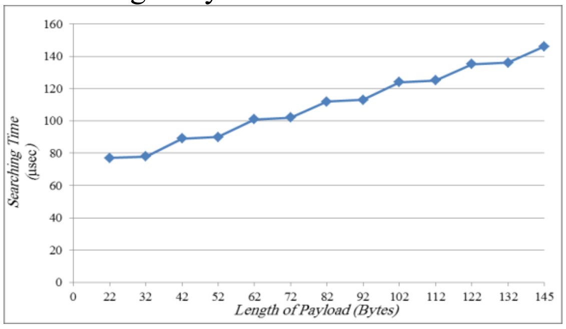


Figure10. The searching time for different sizes of payload

6. Simulation Framework

The efficiency of the proposed distributed IDS system was assessed through exhaustive simulations. The simulation analysis consists of different gateway nodes which will be linked together by using an IP network. In this paper a detailed simulation models of the sensor nodes are built using network simulation package OPNET IT GURU Academic Edition, which does not offer data acquisition as a standardized application, so the simulation procedure adopted in this paper depends on the procedure explained in [15] which developed an efficient way to simulate field devices. The Wi-Fi networking model is built by creating projects and worked on the model at the OPNET's network layer. The simulation analysis includes two case studies.

6.1 Case Study 1

In the first scenario, we simulated the proposed WSN gateway armed with IDS as shown in **Fig.11.** Depending on the proposed system described in section 3, the simulation scenario consists of:

- Two IDS Sensors (IDS_Sensor1 & IDS_Sensor2) which are specific PCs in which *SNORT* program was installed on them (with a complete rules set) and located in different locations in the network.
- Classification and Processing Server which collects the reports from IDS sensors and analyzes them to assign the most common risky attacks with high priority.
- Five WSN gateway nodes armed with IDS which are used to sense particular event in industrial environment when receives notification from classification and processing server.

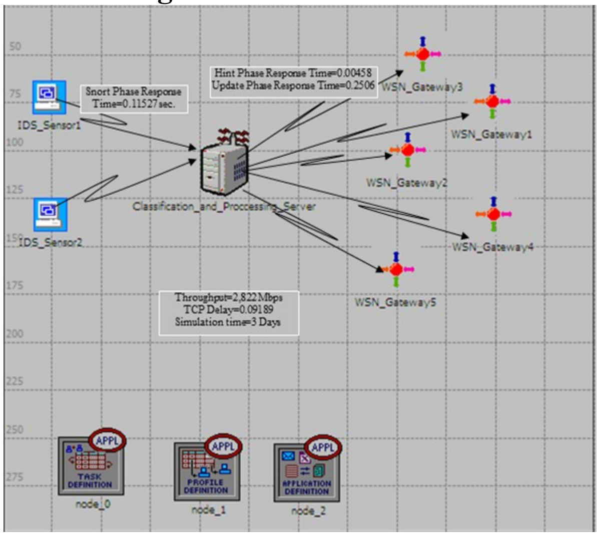
The following assumptions where considered during the simulation:

- The Wi-Fi network model is assumed to work with IEEE.802.11b standard with a maximum speed of 11Mbps.
- All the nodes are not dynamic (immobile).
- The dimension of the network is 300*300 m which supposed to be reasonable area in an industrial environment.

For better analysis, a custom application is designed (or created) to analyze the behavior of the proposed system. The flow actions of the application are as flows:

- ❖ *Phase1(Snort Phase):* IDS Sensors send a file of size 10 KB each 1 hour (Snort Rules Updated File) to the Classification and Processing Server.
- ❖ *Phase2(Hint Phase):* After receiving the Snort Rules Updated File, Classification and Processing Server processes and converts the data to produce the rules and sends an update notification of size 1 KB to the Wireless gateway nodes every 1 hour.
- ❖ *Phase3 (Update Phase):* If Wireless gateway nodes respond to the notification, the Classification and Processing Server will send the updating File with a size of 50 Kbps.

The simulation results for this scenario are shown in **Fig. 11**:



Snort Phase Response Time=0.11527 sec.

Hint Phase Response Time=0.00458
Update Phase Response Time=0.2506

Throughput=2,822 Mbps
TCP Delay=0.09189
Simulation time=3 Days

Figure11. Phases' response time

6.2 Case Study 2

The second scenario aimed to validate our development in a Real-Time (RT) environment (industrial environment as an example). A subnet is created to represent an industrial wireless network .We creates a network contains of five gateway nodes (each one is responsible for gathering measured data from many sensor nodes and bridging them to other nodes) armed with an embedded Intrusion Detection System and one controller-actuator node in ad hoc mode. By RT communication, we mean small-sized packets generated in periodic intervals must be delivered before the end of the message stream period. These real-time periodic data exchanges are intended to model both sensor messages sent to controller-actuator node, and output messages sent from controller-actuator node to the sensors via gateways. In this scenario we assumed that the packet processing speed of gateway nodes is 2000 packet/sec (which reflects CPU speed of =120 MHz that simulates the frequency of UBICOM network processor), packet length is 100 byte (small-sized packet length) and packet production rate is 200 packet/sec. this scenario tested the effect of non-real time data on the system by assuming that one gateway node receiving a file of 50Kbyte size (as Rules updating File)using file transfer protocol (FTP protocol).

Fig. 12 shows clearly the effect of packet end-to-end delay. It's noted that the presence of non-real time data (Rules updating file) in the system has a negative effect on the system performance which increase the delay value due to the additional load added to the system. In our future work, care will be paid to overcome this problem using Quality of Service(QoS) techniques.

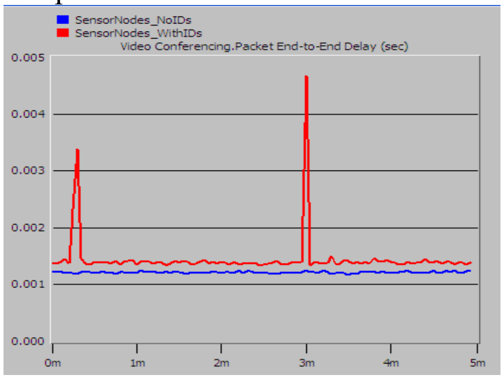


Figure12. Packet end-to-end delay

7. Conclusions

Networked sensor systems are seen by observers as an important technology that will experience major deployment in next few years for a variety of applications. Adding different security methods is an important and essential procedure to enhance the operation and safety of such system. This paper focused on the design and implementation challenges to localize an embedded version of SNORT Intrusion Detection System into wireless gateway nodes. Our design takes into account the "embedded" nature of the gateways and their limited resources and suggests different methods and protocols to achieve its goals. The obtained results prove the possibility to insert IDS functionality in the system with minimum effect of its normal operation.

8. References

[1]. Safiqul I., Razib H., Dewan M., "A Hierarchical Intrusion Detection System in Wireless Sensor Networks", IJCSNS International Journal of Computer Science and Network Security, VOL.10 No.8, August 2010.

[2]. Chong E., Mun Y., Christopher L., Marimuthu P., "Intrusion Detection for Routing Attacks in Sensor Networks", NICTA Victoria Laboratory , Department of Electrical and Electronic Engineering, the University of Melbourne Parkville, Victoria, Australia, 2005.

[3]. Giannetsos A., "Intrusion Detection in Wireless Sensor Networks", Master thesis, Mellon University, April 8, 2008.

[4]. Manju. V.C, "Study of Security Issues in Wireless Sensor Network", International Journal of Engineering Science and Technology (IJEST), Vol. 3, No.10, October 2011.

[5]. Teodor-Grigore L., "Main Types of Attacks in Wireless Sensor Networks", Department of Computer and Software Engineering, University "Politehnica" of Timisoara, Faculty of Automatics and Computers, Timisoara, Romania, 2008.

[6]. Hichem S., Mohamed F., "Novel Hybrid Intrusion Detection System for Clustered Wireless Sensor Network", International Journal of Network Security

& Its Applications (IJNSA), Vol.3, No.4, July 2011.

[7]. Andriy S., Lukas F., Vaclav M., "Neighbor-Based Intrusion Detection for Wireless Sensor Networks", Faculty of Informatics Masaryk University (FI MU), Report Series FIMU, FIMU May 2010.

[8]. Theodore Z., Panagiotis T., Helen L., Kostas P., Evangelos L., Christos T., Charalampos V., Lionel B., Jukka M., Michalis L., Federico A., Yannis P., "Securing Wireless Sensor Networks Towards a Trusted "Internet of Things"", Towards the Future Internet G. Tselentis et al. (Eds.) IOS Press, 2009.

[9]. Ioannis Ch., Andreas S., "A Decentralized Intrusion Detection System for Increasing Security of Wireless Sensor Networks", supported by the IST Program of the European Union under contract number IST-2005-15964 (AEOLUS).

[10]. Timothy K., Faiz M., Randy C., Joseph G., "Battery-Sensing Intrusion Protection for Wireless Handheld Computers using a Dynamic Threshold Calculation Algorithm for Attack Detection", Proceedings of the 40th Hawaii International Conference on System Sciences, 2007.

[11]. Shio K., Singh M., Singh D., "Intrusion Detection Based Security Solution for Cluster-Based Wireless Sensor Networks", International Journal of Advanced Science and Technology Vol. 30, May 2011.

[12]. Aldwairi M., "Hardware -Efficient Pattern Matching Algorithm and Architectures for Fast Intrusion Detection", PHD. Dissertation, Computer Engineering Dept., North Carolina State University, 2006.

[13]. Pankaj Gupta, "Algorithms for Routing Lookups and Packet Classification", PHD. Dissertation, Department of Computer Science, Stanford University, December 2000.

[14]. "IP2022 Wireless Network Processor Features and Performance Optimized for Network Connectivity IP2022 Data Sheet", UBICOM, Inc., 22 Jan. 2009, Web Site: http//www.ubicom.com.

[15]. Q. I. Ali, "An Efficient Simulation Methodology of Networked Industrial Devices", IEEE SSD08 Conference, JORDAN, 2008.