Security Architecture for Resource-Limited Environments

S. Nair, S. Abraham, and O. Al Ibrahim
SMU HACNet Labs
Southern Methodist University
Dallas, TX, USA
{nair, smabraham, oalibrahim}@smu.edu

***Abstract*— Security in resource-limited environments poses a great challenge as the nodes comprising such a network are severely limited in processing power and storage. Various security protocols have been proposed for sensor networks and RFID that try to mitigate this problem but they provide security on a point-to-point basis which is weak and easily vulnerable to a number of attacks. In this paper, we discuss architecture for a resource limited environment that enables these networks to achieve better security and reliability while being simple and efficient. Based on the architecture we develop non-cryptographic security mechanisms using state machines for providing basic security services including authenticity and confidentiality. We also introduce the concept of Security Fusion where strong system level security properties are synthesized from weak point-to-point security properties. The strengths of these techniques are analysed so as to offer comparisons with conventional techniques.**

***Keywords* — Sensors, RFID, security**

I. INTRODUCTION

Resource-limited devices such as sensors and RFID are being widely deployed in a host of different environments to monitor and protect critical infrastructures. For example sensors have gained a lot of popularity in recent times due to some of its natural advantages including small size, reduced cost, and potential for deployment [5, 6, 10, 12].

Although resource constrained devices such as sensors and RFID can provide promising solutions for a number of different applications, concerns about security have been the stumbling block to the deployment of these devices [5]. Providing security for these resources is very challenging for various reasons. First, since these devices have limited computation, memory, and energy resources, it is difficult to employ strong cryptographic solutions used in traditional networks for such weak nodes. For example the popular encryption algorithms such as the RC4 implemented in the TinySec architecture uses a key size that is less than 64 bits in length. With current advances in technology, it is possible to complete an exhaustive key word attack on such networks without much difficulty. Moreover, as new emerging proposals for the deployment of MEMS and nanotechnology based sensors gain widespread acceptance, it is not practical to migrate these cipher algorithms to run on such a significantly resource limited hardware. Secondly, these devices use wireless communication, making it susceptible to eavesdropping. In addition, an attacker can easily inject malicious messages into the wireless network. Finally, they are susceptible to physical capture and providing tamper-resistant hardware is not an option due to their target cost. Consequently, an attacker may compromise nodes easily.

In this paper, we present a novel security framework for resource-limited environments. We make the following contributions.

- We propose new security architecture for systems severely limited in processing power and storage.
- We analyze the security characteristics of our proposed scheme.
- We evaluate the resiliency of our proposed security scheme against different types of attacks.
- We introduce the concept of security fusion using the proposed security framework as the building block.

The remainder of the paper is organized as follows. In section 2, we discuss previous research, followed by our proposed security architecture for secrecy and authenticity, in section 3. Next, we provide an in-depth analysis of our proposed architecture in section 4. Finally, we provide conclusions in section 5.

II. PREVIOUS RESEARCH

Researchers have investigated the challenges in securing resource constrained nodes such as wireless sensor networks while also maximizing their computational capabilities and energy utilization. Various methods have been proposed to protect sensor networks from attacks. Mainly, standard cryptographic techniques have been used to protect the secrecy and authenticity of communication links from various attacks. Perrig et al. [1] introduced SPINS which has two building blocks: SNEP- a security protocol – that employs symmetric cryptography using RC5 for encryption and Message Authentication Code (MAC) and µTESLA which provides authentication for data broadcast. Wagner et al. [2] described TinySec, a link layer security architecture for micro sensor networks.
Researchers have also begun focusing on using secure information aggregation techniques to protect data that has

978-1-4244-9538-2/11/$26.00 ©2011 IEEE

been aggregated from multiple nodes in the network. There has been considerable research on the techniques of Sensor Data fusion but it is based on the assumption that all nodes are trustworthy. This rule doesn't hold true in a sensor network environment and so it is necessary to secure the aggregated data. In [31], Perrig et al proposes a novel framework for secure information aggregation in sensor networks. They use a three phase approach also known as aggregate-commit-prove to verify the correctness of the results aggregated from multiple nodes and rejects results in the event that the nodes are cheating. However the focus of this paper is to protect the data that is fused from multiple sensor nodes. It doesn't secure the individual values nor is the backend able to extract the values emitted by each node.

III. Security Architecture

a) System Model:

We envisage that these types of sensors will be deployed in groups of large number of nodes. Electronically, the networks will have the architecture shown in Figure 1.The network is a cluster based system, with one cluster head (reader) belonging to a cluster as shown in Figure 1. The nodes cannot communicate among themselves. However, they can communicate with the reader. The reader initiates communication and may periodically interrogate the nodes.

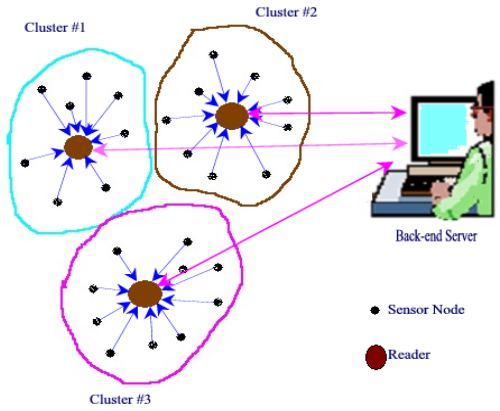

Figure 1: Sensor Network Architecture

The sensor nodes are powered by the reader and equipped with an integrated sensing component, computational component, storage component, and communication component. Our sensor nodes are architecturally similar to a passive RFID tag integrated with a transducer. Each sensor node is associated with a state machine and a transition to a new state is triggered on every read. We propose the use of a finite state machine due to its simple computational model. The reader in each cluster is connected to a back-end server that keeps all the information--- state machine, node_ID--- related to each sensor node. The readers themselves can keep all the information related to each sensor node belonging to the corresponding cluster. The communication channels between the cluster heads (readers) and the back-end server are secure.

b) Security Protocol:

Although our proposed security protocols can be used for various applications, for simplicity, below we describe the protocols considering a scenario where only binary information (0/1) is being collected from the sensor node. For example 1 could represent the presence of a harmful chemical (detection) and 0 could represent its absence (non-detection).

Each sensor node is associated with a unique state machine. After each interrogation from the reader, the nodes may change their states according to the transition rules defined prior to deployment. In addition, each state in the state machine is assigned k numbers or *pseudonyms*, of which p ($1 \leq p < k$) numbers may be used to represent a detection (1) , and q (q=k-p) numbers may be used to represent a non-detection (0). Furthermore, a unique ID is assigned to each sensor node for identification purposes. The state machine for each node is shared with the decision maker (cluster head or/and back-end server) only. Prior to deployment, the decision maker is provided with each node's initial state. If a sensor node changes its state, so does the decision maker, to remain in-sync.

Denote ***S***: sensor node, ***R***: reader
The Basic Protocol

1. *R* → *S:* Send read query
2. *S:* Obtain <*sensed value*> *(0/1)*
3. *S* → *R: S* moves to the next state based on <*sensed value*> and outputs an pseudonym
4. *R* resolves *S*'s output and syncs

When interrogated by a reader, a sensor node responds with its ID and a number, which may depend on the detection or non-detection, node ID, and its current state. Since the decision maker, while interrogating a sensor, knows the node's state and the k numbers assigned for that state, it could simply authenticate a sensor node by the number. The decision maker can also deduce actual information (detection or non-detection), using the mapping between the numbers and the actual information. After getting the response, the decision maker will also update its copy of the state machine corresponding to that node. In addition, data confidentiality is also achieved as the sensor node sends the numbers only from which unauthorized parties cannot deduce the actual information without the mapping between the numbers and the actual information, and the mapping is never transmitted over the air. In other words, the transitions and the mappings in a state machine represent the secret key that is shared between that particular node and the backend server. Since the state machine transitions upon every interrogation, the secret key (transitions & mapping) will change periodically during the lifetime of the network which protects against replay attacks.

Collectively, the decision maker authenticates a response pattern according to a consensus criterion, which could be such that 80% of the node should get authenticated. Only if the consensus criterion is met, does the decision maker apply a majority logic—a fusion algorithm-- on the responses from the nodes to make a decision regarding the 'detection" or "non-detection". If the consensus criterion is not met, then the decision maker discards all the responses.

c) Formal Description of the State Machine:

In this section we provide a formal description of the state machines based on "output" and "transition" rules. For the sake of simplicity, we are considering a sensor node that collects only binary information (0/1). In this case "detection" of a harmful chemical substance could represent a 1 and non-detection could represent a 0. Depending on detection or non-detection, the output rules provide a mapping between the sensor node's current state and the numbers that need to be transmitted by a sensor node. Let us assume that we have state machines with ***n*** states ($S_1, S_2, \ldots, S_n$) and the following rules:

Transition rules define transition from current state to next state based on the input values:

(Current state, Input) → Next state

1) *(S_i, "detection/1") → S_j*
2) *(S_i, "non-detection/0") → S_v, where (0< i, j, v≤n)*

Output rules: (Current state, Input) → Number

1) *(S_i, "detection/1") → a_i*
2) *(S_i, "non detection/0") → b_i, where $a_i \neq b_i$.*

The output rules define the possible values (*pseudonyms*) emitted by the device for a given state and transition. Hence the value (0/1) obtained by the sensor is mapped to one of the k pseudonyms associated with a state. We will step through several Observation results to capture the total number of possible state machines that can be generated given the number of states and pseudonyms.

Observation 1: With n states, each of which may move to any other states including itself depending on the two inputs, the number of state machines that could be generated is $(n)^{2n}$.

Observation 2: The number of ways of partitioning a set of n objects into r cells with n1 elements in the first cell, n2 elements in the second, and so forth, is

$$\frac{n!}{n_1!n_2!.....n_r!} \tag{1}$$

where $n_1 + n_2 + \ldots\ldots + n_r = n$.

Observation 3. If each state is assigned a set of k numbers, of which p (1≤ p < k) numbers may be used to represent a detection, and q (q=k-p) numbers may be used to represent a non detection, then the number of possible ways that we can assign numbers to a state (using Observation 2) is:

$$= \sum_{p=1}^{k-1} \frac{k!}{p!(k-p)!} \tag{2}$$

Observation 4: The number of ways of partitioning a set of nk numbers into n states with k elements in each state is

$$= \frac{nk!}{(k!)^n} \tag{3}$$

Observation 5: With n states, each of which may move to any state depending on two input values, and with nk numbers to be assigned into n states with k elements in each state, of which p (1≤ p < k) numbers may be used to represent a detection, and q (q=k-p) numbers may be used to represent a non detection, the total number of possible state machines that could be generated (using Observation 1, Observation 3 and *Observation 4*) is:

$$n^{2n} \left[\sum_{p=1}^{k-1} \frac{k!}{p!(k-p)!}\right]^n \left[\frac{nk!}{(k!)^n}\right] \tag{4}$$

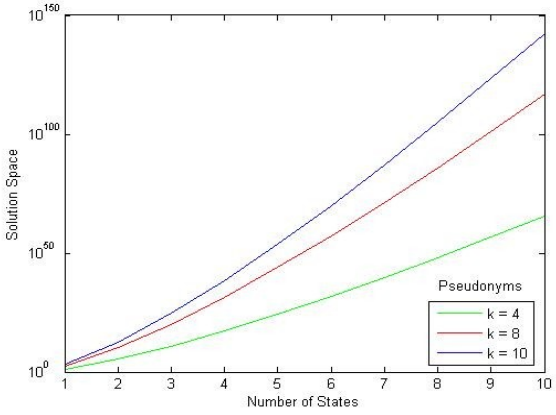

Figure 2: Solution space using Observation 5

Figure 2 represents solution space for generating every possible state machine having 1 to 10 states. From this figure, we observe an exponential expansion in the key space by increasing the key size in two dimensions: the number of states *n*, and number of pseudonyms *k*. It is to be noted that in ***Observation 1*****,** we have also included the set of all state-machines that are connected and disconnected. So we need to discard the state-machines that are disconnected and only keep the ones that are connected. This is accomplished by performing a connectivity test on each state-machine that is

generated. A state-machine is an instance of a directed-graph. Hence for each state-machine derived using ***Observation 1***, we will test it using a connectivity algorithm to eliminate any disconnected state machines. There is a known list of algorithms [28], [29] that can perform this task in optimal time.

d) State Minimization:

Since our protocol doesn't give out the actual state value, intruders can collect only the numbers assigned to the states. As a result, an intruder can derive an equivalent Non-deterministic (NFA) state machine using those numbers. However, in order to derive the original state machine, the intruder needs to minimize the states. Since the number of states in a state machine is n, and k values are assigned to each state, the number of states in the state machine derived by the intruder will be nk. Consequently, the complexity of deriving the original state machine would be $O(nk\log nk)$ [19]. We will illustrate this with an example.

Consider a 3-state Finite State Machine (FSM)

1. **n=3** {s1, s2, s3}. Each state *(s1, s2, s3)* will have k numbers assigned to it.

2. **k=3** [Each state is assigned a set of 3 numbers of which p (1<= p < k) numbers may be used to represent detection (1) and q = k-p numbers may be used to represent a non-detection (0).]

3. The total set of numbers available for the 3- finite state machine = **nk** = 9

Based on Observation 5, we can have approximately 2.64 x 10^8 different instances of state-machines generated using the above configuration. A depiction of one particular instance of a state machine is shown in Figure 3. As shown in the figure, there are two possible transitions from a state based on detection (1) and non-detection (0). For state S1, on non-detection it will transition to state S2 and emit either 1 or 2. The transition tables and output rules for this state-machine is shown in table 2. The back-end server also keeps a copy of this state-machine instance and keeps it in sync with the sensor-node.

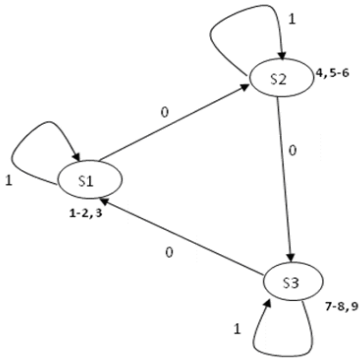

Figure 3: State diagram

States	Transition on "0"	Transition on "1"
S1	1, or 2	3
S2	4	5, or 6
S3	7,or 8	9

Table 2: Pseudonym assignment

Since the node doesn't provide the actual state values, the attacker can only collect information about the numbers k associated with each of the 3 states. Hence the number of states derived by the intruder will be nk. From these nk values and by observing the transitions based on detection (1) and non-detection (0), the intruder will be able to derive an equivalent state-machine with 9 states. When the value of k is greater than 2 (k>2), the intruder will always derive a finite state machine that is non-deterministic in nature.

Based on Hopcroft's Algorithm [9], it takes *nlogn* steps to minimize a deterministic finite state machine with n-states. Hence if the state machine derived by the intruder were deterministic, it would take *(nk)log(nk)* steps to derive the original state machine. However since the state machine derived by the intruder is non-deterministic (NFA) in nature, the intruder will have to convert it first to a Deterministic Finite Machine (DFA) first before applying the minimization algorithm (Hopcroft's Algorithm). Based on [30] converting an NFA to a DFA can cause an exponential increase in the number of states. That is, if an NFA has *m* states, then the resulting DFA may have 2^m states. This exponential increase makes the problem of minimization NP-hard whenever you do not start with a DFA.

IV. ANALYSIS

For the security architecture considered in this paper, one of the goals of attackers would be to find out the secret (transition rules and the output rules) shared by the nodes and the decision maker. Once the secret is known, attackers will be able to force the decision maker to make a wrong decision, by providing false information on the sensed data. Moreover, by knowing the output rules alone, an attacker can determine the actual information just by eavesdropping responses from the nodes.. Below, we have identified a number of attacks and evaluated our protocol against those attacks.

a) Passive brute-force attack:

In a passive attack, an intruder cannot actively participate in the protocol. Intruder can only eavesdrop. By eavesdropping a conversation between the reader and a node, an intruder may collect packets that contain Node_ID and the output numbers, and then the intruder can attempt to determine the secret (transition rules, mapping) shared by the node and the reader. However, in order to find all the transition rules associated with a node, the intruder needs to collect packets containing

all the *nk* numbers. Once, all the numbers have been collected, the intruder can derive the transition rules by looking at successive packets. In order to get all the *nk* numbers assigned to each state machine, the attacker would have to collect a significantly large amount of packets. The following equation *(Equation 1*) provides an estimate of the number of packets that will need to be read from the sensor-node based on 90% probability to determine all the nk values associated with the state-machine.

$$F(r)=[\sum^{nk}_{p=0}(-1)^{p}C(nk,p)*(nk-p)^{r}] / (nk)^{r} = 0.9 \rightarrow Eq.\ 1$$

(The coefficients in front of the powers come from Pascal's triangle and r represents the number of packets read.)

For example, if we have a state-machine with two states and each state having 3 values (i.e. n = 2 and k = 3), then the attacker can be 90% sure that he can determine all the nk or 6 values associated with the state machine by reading at least 23 packets. Thus the complexity for an intruder to gather all nk values to derive the original state machine is increased significantly even if k is increased moderately. The above analysis assumes that the probability of emitting any one of the *nk* values associated with the state machine is the same. In order to have state-machines with such properties we can apply a randomness test on the state machines generated from Observation 5. Each of the state-machines will be provided with an input containing a random stream of 1's & 0's (about 100000 samples). And based on the output numbers generated by the state machine, a frequency analysis will be performed on them. If the frequencies are very unevenly distributed, then the state-machine is discarded. This will enable us to work with state-machines that are truly random in nature. More importantly, since our protocol depends on the correct information from a global secret, and the complexity of deriving the global secret would be an *NP* hard problem.

b) Active brute-force attack:

Complexity of a brute-force attack depends on the amount of information that the attacker already has. Based on the Kerckhoffs' law of security protocol design we assume that an intruder has knowledge of the underlying protocol used including the number of states *n* and the set of numbers used in the output mapping (Note that this is reasonable assumption since the intruder can eavesdrop on several read cycled to get this information). However, he/she doesn't have access to the mapping and the actual transition rules as they are the secrets.

In an attempt to spoof responses the intruder may replicate a sensor node by randomly picking transition rules and output rules. After replicating a node, the intruder may insert it to the network, hoping to disrupt the system. Here spoofing responses means that a replicated node is giving incorrect value for the sensed data, which will be authenticated by the decision maker. Note that the intruder needs to pick the right transition rules and the output rules to provide false responses. Number of combinations the intruder may have to use is large as indicated in the Observation 5 provided along with the formal description of the protocol. Replication of falsification for all the nodes in the system will further increase the complexity. We can prove that the Non-Deterministic finite state machine derived by the attacker will have a total of approximately nk^2 edges or transitions in the state machine. If the attacker has the means to replicate multiple copies of a given sensor node then the complexity to derive all the transition rules or edges of the state machine would be $O(nk^2)$. However if this is not the case which is the most likely scenario, then the number of transitions the attacker will have to detect to derive the secret based on a 90% probability is:

$$F(r)=[\sum^{nk^{\wedge}2}_{p=0}(-1)^{p}C(nk^{2},p)*(nk^{2}-p)^{r}] / (nk^{2})^{r} = 0.9 \rightarrow Eq.\ 2$$

(The coefficients in front of the powers come from Pascal's triangle and r represents the number of packets read.)

c) Physical Attack:

A determined intruder may go as far as to capture sensor nodes and then find out the secrets shared between the nodes and the decision maker. However, the number of nodes that could be captured will depend on a number of factors including accessibility of the sensor field and location of the individual nodes. By capturing a sensor node, an intruder may find the transition rules and the output rules associated with that sensor node. However, compromising a node will not give out any secret from other nodes as no two nodes are associated with the same state machine. Further note that to derive the global secret on which our security protocol depends, an intruder needs to capture a significant number of nodes.

d) Malicious Read:

With a rogue reader in possession, an intruder may participate in the protocol by making an attempt to read out from the nodes with the rogue reader. Since there is no mechanism in the proposed architecture to authenticate a reader, anybody with a reader can read out from a node. By interrogating the nodes repeatedly, the intruder may collect packets and try to derive the state machines. However, if the attacker does not have access to actual value of the sensed data during interrogation, then the intruder can not determine the output rules.

e) Replay Attack:

In this attack, an intruder may collect packets and replay those packets at a later time. However, since a node changes its mapping (unless it does not change states) after every response, and it is authenticated by its current mapping, an intruder will find it difficult to launch a replay attack. In order for a replay attack to be successful, an intruder must know the state machine shared by a node and the reader. Otherwise, the decision maker will consider it as an unauthenticated response.

f) De-synchronization Attack:

An attacker may try to upset the synchronization between the decision maker and the nodes by interrogating the nodes with a rogue reader. Again, since the sensor nodes are not equipped with any mechanism to authenticate a reader, if the attacker attempts to read out from a node, the node will give out the output, and at the same time will change its state according to the transition rules, leaving the nodes and the trusted decision maker out of sync. However, a legitimate reader will detect that the nodes have been read out as the reader and node state machines will be out of sync. The decision maker can realign itself with the nodes by advancing itself to desired state through several transitions. One possible approach to speed up the re-sync is to send two readouts that are processed atomically by the node. In response, the node emits two pseudonyms: one from the current state and one from the next state. Knowing the mapping rules, the system can deduce the states and re-sync itself with the node.

V. Conclusions and Future Work

In this paper, we presented security architecture for a resource limited environment that comprises of nodes such as RFID and sensors that have limited processing power, storage, bandwidth, and energy. Since standard cryptographic solutions are likely to be impractical for such an infrastructure, our architecture employs non-cryptographic techniques based on multiple-valued state machines to provide security and reliability. Looking at security from a system/global point of view, we have presented an in-depth analysis of our architecture. Work is underway to validate our security model and to compare our solutions with standard cryptographic techniques. In addition, we plan to extend this research into areas that support the concept of "security fusion". Our focus will be on improving the overall security of resource-constrained infrastructure through the synergistic effect of security data obtained from multiple nodes. Our study will include techniques to build a security system based on fusion across different security attributes, across domains and across time where we will use environmental data and a priori data to make inferences about the overall security of the environment.

VI. References

[1] Perrig, R. Szewczyk, V. Wen, D. Culler, J. D. Tygar. "SPINS: Security protocols for sensor networks." Proceedings of Seventh Annual International Conference on Mobile Computing and Networks, July 2001.

[2] C. Karlof , N Sastry., D. Wagner. "TinySec: A link layer security architecture for sensor networks". In Proceedings of the Second ACM Conference on Embedded networked Sensor Systems (SenSys), November 2004.

[3] A. Madhukar, I. Zachary, L. Insup; "Quantifying eavesdropping vulnerability in sensor networks" Proceedings of the 2nd international workshop on Data management for sensor networks DMSN '05 ; August 2005.

[4] L. Escenauer, V. D. Gilgor. "A key-management Scheme for Distributed Sensor Networks." Proceedings of 9th ACM Conference on Computer and Communication Security, Nov 2002, Pages: 41-47.

[5] E. Shai, A. Perrig. "Designing Secure Sensor Networks." IEEE Wireless Communication. Volume: 11, Issue: 6, Dec 2004, Pages: 38-43.

[6] C. Karlof, D. Wagner. "Secure routing in Wireless Sensor Networks: Attacks and Countermeasures." Proceedings of the 1st IEEE Workshop on Sensor Network Protocols and Applications, May 2003, Pages: 113-127.

[7] J. Newsome, E. Shi, D. Song, A. Perrig. "The Sybil Attack in Sensor Networks: Analysis and Defenses." Proceedings of IEEE International Conference on Information Processing in Sensor Networks. April 2004, Pages 259- 268.

[8] A. Wood, J. Stankovic. "Denial of Service in Sensor Networks." IEEE Computer, Oct 2002, Pages: 54-62.

[9] W. Du, J. Deng, Y. S. Han, S. Chen, P. Varshney. "A key management scheme for wireless sensor networks using deployment knowledge." Twenty-third Annual Joint Conference of the IEEE Computer and Communication Societies, Volume: 1, March 2004. Pages: 586- 597

[10] C. Chee-Yee, S. P. Kumar. "Sensor Networks: evolution, opportunities, and challenges." Proceedings of the IEEE, Volume: 91, Issue: 8, Aug 2003, Pages: 1247- 1256.

[11] H. Chan, A. Perrig, D. Song. "Random Key Pre-distribution Schemes for Sensor Networks." IEEE Symposium on Research in Security and Privacy. May 2003.

[12] L. Gasman. "Thoughts on the Economics of Nanosensors." A white paper published by Nanomarkets in Dec 2004.

[13] A. Modi, N. koratkar, E. Lass, B. Wei, P. Ajayan. "Miniaturized Gas Ionization Sensors using Carbon Nanotubes," Nature, Vol. 424, Jul. 10, 2003, pp. 171-174.

[14] O.K Varghese, D. Gong, M. Paulose, KG Ong, and C.A Grimes. "Hydrogen sensing using titania nanotubes," Sensor and Actuators B, volume: 93, 2003, Pages: 338-344 .

[15] J. Kong, N. R. Franklin, C. Zhou, M. G. Chapline, S. Peng, K. Cho, H. Dai. "Nanotube Molecular Wires as Chemical Sensors," Science, Volume: 287, Jan. 28, 2000, Pages: 622-625.

[16] Nanomix

[17] E. Smalley. "Chip Senses Trace DNA." Technology Research News, Jul 30/Aug.6, 2003.

[18] Ultralight device analyzes gases immediately. Flying SnifferSTAR may aid civilians and U.S. Military," Sandia National Laboratories, Press Release, Jan. 23, 2003.

[19] G. De Micheli. "Synthesis and Optimization of Digital Circuits." McGraw-Hill, Inc., 1994.

[20] J. Farnendez; A. Farnendez. "SCADA Systems : vulnerabilities and remediation." Journal of Computing Sciences in Colleges, Volume:20, Issue:4, April 2005

[21] E. Luiijf; M. Klaver "Protecting a Nation's Critical Infrastructure: The First Steps" IEEE International Conference on Systems, Man, and Cybernetics. 2004

[22] "Critical Infrastructure Protection:Challenges and Efforts to secure Control Systems" General Accounting Office (GAO) report GAO-04-354, March 2004,

[23] "Critical Infrastructure Protection: Challenges and Efforts to secure Control Systems" General Accounting Office (GAO) report GAO-04-354, March 2004,

[24] K. Mahmud; S. Nair, "A Security Architecture for Nano-Sensor Networks", Tech Report 06-CSE-02 , CSE Dept, Southern Methodist University, Dallas, Texas, 2006.

[25] A. Miller. "Trends in Process Control Security" IEEE Security and Privacy Magazine. Volume: 3, Issue:5, Sept-Oct 2005, pages:57-60.

[26] G. Cybenko, J. Guofei. "Developing a Distributed System for Infrastructure Protection." IT Professional. Volume:2, Issue:4. Jul-Aug 2000, Pages:17-23.

[27] K.B. Lee, M.E. Recichardt. "Open Standards for Homeland Security Sensor Networks." IEEE Instrumentation & Measurement Magazine, Volume: 8, Issue: 5, Dec 2005, Pages: 14-21.

[28] Esfahanian, A.H., Hakimi, S.L., On computing the connectivities of graphs and digraphs, Networks, 14 (1984), pp. 355-366.

[29] Matula, David and Vohra, Rakesh Calculating the connectivity of a directed graph - Institute for Mathematics and its applications - Jan 1988

[30] "The Algorithm Design Manual - Steven S. Skiena" Published by Springer, 1998 page:419

[31] B. Przydatek,D. Song,A. Perrig. "SIA: Secure Information Aggregation in Sensor Networks"