International Journal of Control and Automation
Vol.7, No.9 (2014), pp.55-66
http://dx.doi.org/10.14257/ijca.2014.7.9.05

A Combined System of Secure Hashing and Neural Networks in Sensor Networks of Living Environment

Sang-Hyun Lee[1], Sang-Joon Lee[2] and Kyung-Il Moon[1]

[1]*Dept. of Computer Engineering, Honam University*
[2]*School of Business Administration, Chonnam National University*
{leesang64, kimoon}@ honam.ac.kr, s-lee@chonnam.ac.kr

Abstract

Sensor networks have a significant potential in diverse applications, and some have already been deployed in monitoring system of living environment. With the increasing complexity of application logic, difficulties in monitoring sensor networks have become a barrier to the adoption of these networks. The difficulties are due not only to their inherently distributed nature but also to the need for mechanisms to address their harsh operating conditions such as unreliable communications, faulty nodes, and extremely constrained resources. Living Environment monitoring is composed mainly of sensor data on air, water, and ecotourism quality. Wireless sensor networks(WSNs) entail a substantial loss of energy because there is a need for some mechanisms that can select multiple communications in single communications. This kind of merging is called data aggregation. This paper presents a secure and authentication-based approach to the data aggregation of living environment. User authentication is performed using a secure hash algorithm. This paper also introduces a neural network to check bad packet communications over the network. The results indicate that the proposed approach is more reliable and efficient than existing ones.

Keywords: *Wireless Sensor Network, Data Aggregation, Hashing Algorithm, Living Environment, Neural Network*

1. Introduction

With recent technological advances such as sensors, electronics, and computing devices, researchers have paid increasing attention to wireless sensor networks (WSNs), which typically consist of a large number of low-cost sensor nodes that have strictly limited sensing, computation, and communication capabilities[1][2]. Sensors are distributed across various locations and clustered to select a cluster head for each cluster. Sensors send their information to the cluster-head (CH). Using data aggregation techniques, the CH sends results to the sink. This schema saves energy. Many sensors may produce repetitive data through data aggregation, and therefore this schema reduces extra data[3]. Different schemas are explained in [4]. Data aggregation is crucial in that many protocols are used in this technique[5]. Here various signal-processing techniques such as data fusion can be used to obtain more accurate information. Faulty sensors send wrong data to the CH, which in turn send them to the base station. Therefore, wrong information is processed, and this represents a major disadvantage of existing methods.

Sensor networks of living environment represent a collection of sensor nodes that cooperatively send data on variables such as air, water, and ecotourism quality to the base station. Because sensor nodes of living environment are powered by batteries, efficient power use is crucial in using networks for a long period of time [6, 7]. In this regard, there is a need

ISSN: 2005-4297 IJCA

to reduce data traffic within sensor networks of living environment as well as the amount of data sent to the base station. Sensor networks of living environment represent a considerable improvement over traditional sensors of living environment in the following two ways. Sensors can be positioned far from the actual phenomenon (*e.g.*, something known by sensory perception). In this approach, large sensors using some complex techniques to distinguish a target from environmental noise are required. Several sensors performing only sensing can be deployed. The position of each sensor and the communications topology are carefully engineered. They transmit time series of some sensed phenomenon to central nodes where computations are made and data are fused. This paper presents a case in which an artificial neural network is securely performed over WSNs [8, 9]. For this, the paper extends Holenderski *et al.*, [10] decomposition model to facilitate secure computing. The results show that this combined system can be constructed and that its performance is comparable to that of existing traditional data aggregation and authentication algorithms.

2. Wireless Sensor Networks of Living Environment

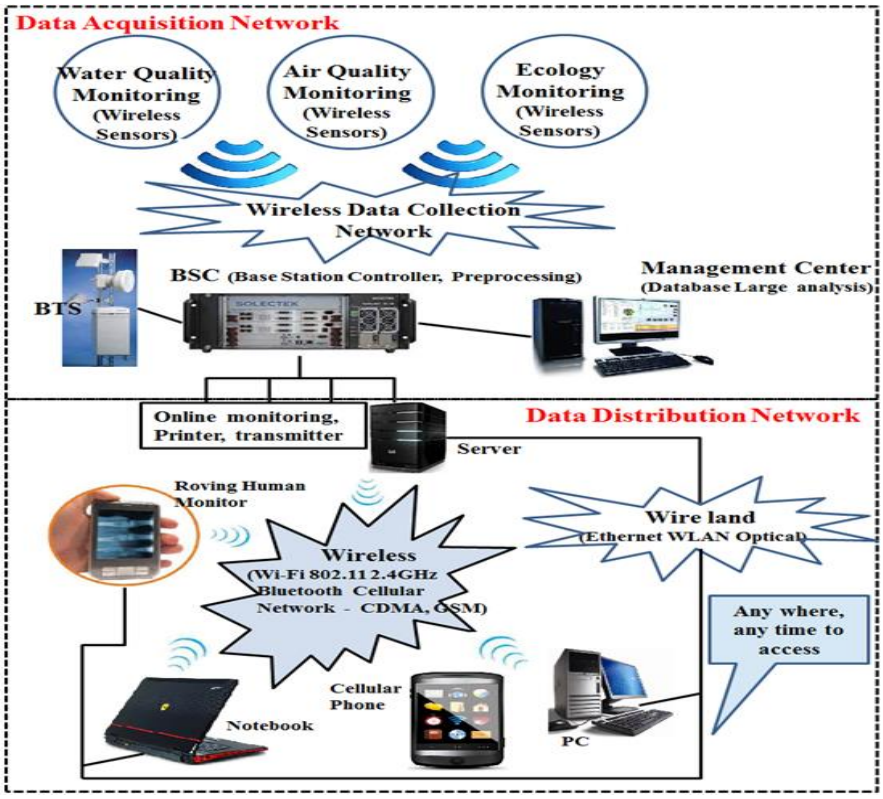

Figure 1. WSNs of Living Environment

A WSN of living environment consists of spatially distributed autonomous sensors to monitor physical or environmental conditions such as air quality, water quality, and eco-tourism quality and cooperatively pass data through the network to the main location. More modern networks are bidirectional, enabling control of sensor activity. The development of WSNs was motivated by military applications such as battlefield surveillance, and such networks have been used for a diverse range of many industrial and consumer applications such as industrial process monitoring and control and machine health monitoring. Sensors

integrated with the structure, machinery, and the environment, together with the efficient delivery of sensor data, can provide the living environment industry with tremendous benefits(see Figure 1).

Sensor data aggregation techniques explore how data are to be routed in the network as well as processing methods applied to packets received by sensor nodes. These techniques have considerable influence on the energy consumption of nodes and thus on network efficiency by reducing the number of transmissions and the length of packets. Fasolo *et al.*, [2] define the in-network aggregation process as follows: "In-network aggregation is the global process of gathering and routing information through a multi-hop network, processing data at intermediate nodes with the objective of reducing resource consumption, thereby increasing network lifetime."

3. Secure Hashing and Neural Networks

SHA-1 produces a 160-bit message digest. The message digest is the fixed-length output of a message. SHA-1 processes a variable-length message into a fixed-length output of 160 bits. The input message is broken into chunks of 512-bit blocks (sixteen 32-bit little-endian integers). This message is padded such that its length is divisible by 512. This padding works as follows: First, a single bit 1 is appended to the end of the message. This is followed by as many zeros as required to bring the length of the message up to 64 bits less than a multiple of 512. Remaining bits are filled with a 64-bit big-endian integer representing the length of the original message (in bits) modulo 264. Bytes in each 32-bit block are big endian, but 32-bit blocks are arranged in little-endian format [4].

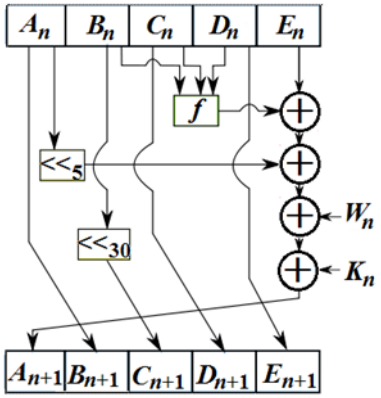

Figure 2. SHA-1

Figure 2 shows iterations within the SHA-1 compression function. Here A, B, C, D, and E denote 32-bit words of the state. f is a nonlinear function that varies; <<t denotes the left bit rotation by t places, where t varies for each operation; Wn is the expanded message word of round t; Kn is the round constant of round t; and ⊕ denotes the addition modulo 232.

The original specification of the algorithm, published in 1993 as the Secure Hash Standard (FIPS PUB 180) by the National Institute of Standards and Technology, a U.S. government standards agency, now often referred to as SHA-0. It has been withdrawn by the NSA shortly after publication and superseded by the revised version published in 1995 in FIPS PUB 180-1 and commonly referred to as SHA-1. SHA-1 differs from SHA-0 only by a single bitwise rotation in the message schedule of its compression function. According to the NSA, this is done to correct a flaw in the original algorithm, namely a reduction in cryptographic security.

However, the NSA provides no further explanation and does not identify the corrected flaw. Since then, weaknesses have been reported in both SHA and SHA-1. SHA-1 appears to provide greater resistance to attacks, providing support for the NSA's assertion of increased security after the change.

Pseudo-code for the SHA-1 algorithm:
// All variables are unsigned 32 bits and wrap modulo 232 when calculating, except
// mj : the message length is 64 bits and
// hd : the message digest is 160 bits.
// All constants in this pseudo code are in big endian.
Within each word, the most significant byte is stored in the leftmost byte position.

// The algorithm starts by initializing five sub-registers
// of the first 160-bit register X labeled h0, h1, h2, h3, and h4 as follows:

h0 = 0x67452301; h1 = 0xEFCDAB89;
h2 = 0x98BADCFE; h3 = 0x10325476;
h4 = 0xC3D2E1F0;

// mj (j=0, 1, … , n-1), the message length in bits (always a multiple of the number of bits in a character).
// SHA-1 iterates through each of the 512-bit message blocks m0, m1, … , mn-1.

for each of the message block
// write mj as a sequence of sixteen 32-bit words:

mj = W0 || W1 || W2 || … || W15.

// Compute the remaining sixty-four 2-bit words as follows:
for (t=16; t<80; t++) {
Wt = (Wt-3 xor Wt-8 xor Wt-14 xor Wt-16) }.

// Copy the first 160-bit register into the second register as follows:
A = h0; B = h1; C = h2; D = h3; E = h4.
// this step involves a sequence of four rounds
// corresponding to four intervals
// 0 t 19, 20 t 39, 40 t 59, 60 t 79.
// each round takes as the input the current value of register X
// and the blocks Wt for that interval and operates on them
// for 20 iterations as follows:

```
for (t = 0; t<80; t++) {
      if (0 ≤ t ≤ 19) {
           f = (B and C) or ((not B) and D)
           k = 0x5A827999
       else if (20 ≤ t ≤ 39) {
          f = B xor C xor D;
          k = 0x6ED9EBA1;}
       else if (40 ≤ t ≤ 59) {
```



```
            f = (B and C) or (B and D) or (C and D);
            k = 0x8F1BBCDC; }
         else if (60 ≤ t ≤ 79) {
            f = B xor C xor D;
            k = 0xCA62C1D6; }
        }
    temp = (A << 5) + f + E + Kt + Wt;
        E = D; D = C; C = B << 30;
        B = A; A = temp.
       }
          // once all four rounds of operations are completed,
          // the second 160-bit register (A, B, C, D, E) is added to
          // the first 160-bit register (h0, h1, h2, h3, h4) as follows:
        h0 = h0 +A; h1 = h1 +B; h2 = h2 + C;
        h3 = h3 + D; h4 = h4 + E.
          // Once the algorithm has processed all of the 512-bit blocks,
          // the final output of X becomes the 160-bit message digest.
```

Sensor networks of living environment consist of a large number of sensor nodes that are typically resource-constrained devices with wireless communications and data--processing capabilities. These nodes collaborate with one another to build communications networks and accomplish their tasks. This paper focuses on distributing supervised learning in an artificial neural network over a WSN of living environment. In a WSN of living environment, there are severe resource constraints, particularly those associated with communications and memory, and these constraints are typically due to low computational power and a short battery life. Therefore, instead of transmitting all sensor data to the sink node and then processing all data in the sink node, distributing supervised learning can be considered as a more effective approach in that the communications cost can be reduced and the computational cost can be distributed across the sensor network.

There are many distributed learning methods. However, most assume ordinary computing environments, not specialized ones such as WSNs. Holdenderski *et al.*, [6] presented a novel model for decomposing sensor nodes to distribute learning work in WSNs and propose two decomposition methods, namely horizontal and vertical ones, to compare computational and communications costs, finding the horizontal decomposition method to outperform the vertical one. However, they do not consider security aspects. Because sensor nodes can be deployed in hostile environments, security is a serious concern in WSNs. Many security breaches or attacking methods have been found, including routing attacks, communications eavesdropping/modifications, impersonation, and denial-of-service(DoS) attacks. In addition, resource constraints and wireless environments make sensor network security more difficult.

Assume that a neural network consists of L fully connected layers with N neurons in each layer. The problem here is to distribute this network evenly over P sensor nodes by assigning to each node a partition of weights and setting corresponding neurons. The cost function to minimize is the maximum communications cost and the memory overhead per sensor node. A neural network of L fully connected layers with N neurons each layer can be divided into as follows: Each sensor node p corresponds to l layers ($l \leq L$) where in each layer p corresponding to u upper neurons, d lower neurons, and all related weights among u and d neurons. If this decomposition is performed with the parameter $l = 1$, then it is called horizontal decomposition. On the other hand, whereas if the decomposition has $l > 1$ and $u = 1$, then it is called vertical decomposition. Because there are N2 weights in each layer, there is

a total of L×N2 weights in the network. If they are distributed evenly across P sensor nodes, then the maximum number of weights assigned to each node is Emax = ⌈LN2/P⌉. If each node p has l layers, u upper neurons, and d lower neurons, then l×d×u≤Emax, which means that, given P, N, L, d, and l, u can be calculated as

$$u \le \lceil E_{\max} /(d \times l) \rceil = \lceil L \times N^2 /(P \times d \times l) \rceil. \quad (1)$$

If l > 1, then some of the partial summation between lower neurons and upper neurons can be calculated by the node p itself, and the receiving communications cost for p is Rx = (l-1)×(N-u)+N if p is not corresponding to the last layer. If the node p is corresponding to the last layer, then Rx = (l-1)×(N-u), where l ≥ 1. As for the transformation of data, because the sensor node p can broadcast messages to all nodes, the transmission cost for p is Tx = l×u if p does not have the first layer. If p has the first layer, then Tx = (l-1)×u. The total communications cost for p is Rx +Tx.

Here the pairwise key establishment can be used for the confidentiality and integrity of communications. A secure routing protocol can be used to defend against routing attacks. To defend against attacks entailing node capturing and incorrect calculations, a voting protocol can be employed. Here, for each weight in the original network, assume that three identical weights are constructed for the new network and layers and neurons are identical to those of the original network. For the decomposition, three different nodes are allocated to vote for a single node in the original/revised protocols. Then the results are compared with each other for confirmation. The receiving communications cost for a secure version is Rx = 3Rx + Rvote. The transmission cost for a secure version is Tx = Tx +Tvote because broadcasting is used for the transmission. The memory overhead cost consists of numbers of weights and neurons: Emax +3(l×max(d, u) + min(d, u)).

4. Results

We begin the task with modeling a wireless sensor network, comprising of ten nodes initially. The scenario containing the sensor nodes along with a base station and router is developed.

Figure 3 shows the simulation results for different network sizes by using MATLAB, and Figure 4 shows the success rate (%).

```
 ===== Network size = 10  maxx = maxy = 57.735 =====
send_app: time 1.002 node 1 sends a crosslayer searching request for key(node) 10
send_app: time 1.004 node 2 sends a crosslayer searching request for key(node) 9
send_app: time 1.006 node 3 sends a crosslayer searching request for key(node) 8
send_app: time 1.008 node 4 sends a crosslayer searching request for key(node) 7
send_app: time 1.01 node 5 sends a crosslayer searching request for key(node) 6
recv_app: time 1.0108 node 3 receives the reply of crosslayer searching with route 3  8
recv_app: time 1.014 node 1 receives the reply of crosslayer searching with route 1   5
recv_app: time 1.0172 node 4 receives the reply of crosslayer searching with route 4  7
recv_app: time 1.0249 node 5 receives the reply of crosslayer searching with route 5  6
timeout_rreq: at time: 1.204 node 2 pending RREQ id=2
timeout_rreq: at time: 1.404 node 2 pending RREQ id=7
timeout_rreq: at time: 1.604 node 2 pending RREQ id=9
timeout_rreq: at time: 1.804 node 2 pending RREQ id=11
timeout_rreq: node 2 has retried so many times to transmit RREQ
--- Network size= 10, Topology id=1, Running time=2.979
```

Figure 3. A Simulation for Different Network Sizes

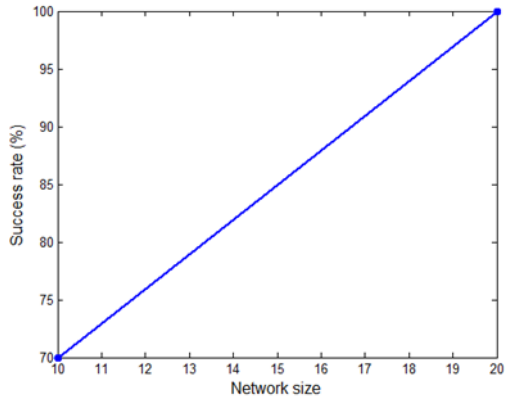

Figure 4. The Success Rate vs. Network Size

Figure 5 shows the response time (sec), and Figure 6 shows the hop count. Figure 7 shows nodes labeled and placed in the WSN. Various nodes are labeled as node 1 to 10. The position of node 1 is fixed, and all remaining nodes are placed in the network by using the "flood" function in MATLAB. The red node indicates the sink node in the network. Figure 8 shows the flow of data across nodes in the WSN.

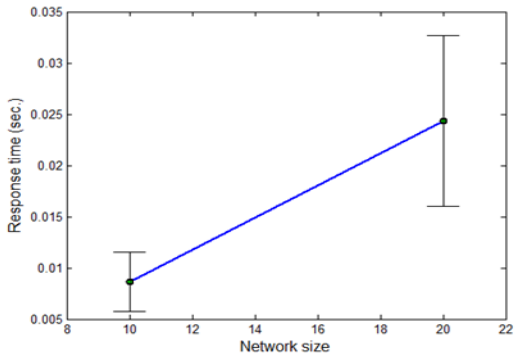

Figure 5. The Response Time vs. Network Size

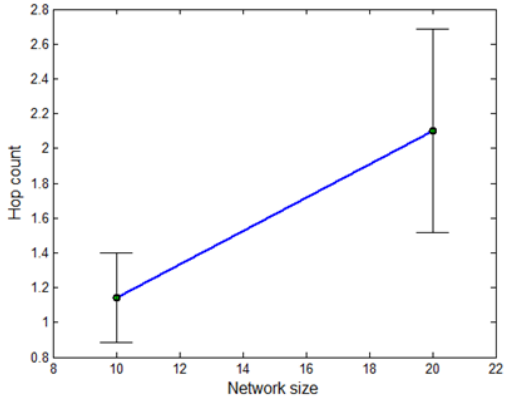

Figure 6. The Hop Count vs. Network Size

Sensor nodes in the modeled WSN use the suggested SHA-1 algorithm for the transmission of data from the sink node to other nodes and vice versa. Once the data transmission is authenticated by implementing the SHA-1 algorithm, a dialog box prompts the user ensuring the authenticity of the data being transmitted among the sink node and other nodes. The keys for transmission are generated and distributed. The neural network tool box is used to train the ANN-WSN network (see Figure 9).

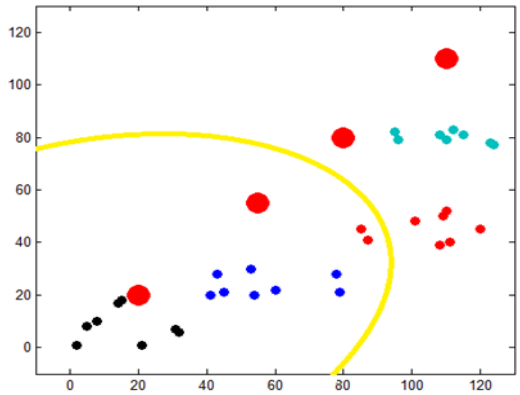

Figure 7. The Modeling of a WSN

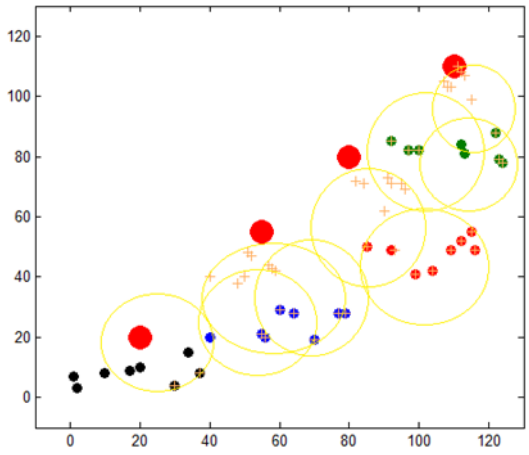

Figure 8. The Data Flow Among the Nodes

The result is reasonable in sense that the final mean square error is small and the test set error and the validation set error has similar characteristics. In addition, there is no significant over fitting by iteration 8 (where the best validation performance occurs). The training phase is related to the validation phase, although the testing phase is different from the other two phases shown during the system's learning process.

Validation and test curves are similar. As shown in Figure 10, the graph suggests that the constructed neural network begins learning at the 14-th epoch. There are six validation checks at epoch 14. One of the graphs is plotted by taking the Val Fail value and the number of epochs. In this graph, the ANN-WSN didn’t perform any learning until the simulation reach epoch number 8. After the epoch 8, the neural network starts learning linearly throughout the

end of the simulation. The other graph shows the gradient variation along with number of epochs. The graph shows an S-shaped curve to the gradient value of 0.009235. This indicates the variation along the S-shaped curve shown in the training state. Given the performance of the neural network, it can be concluded from the plot that the neural network reduces the detection of false data better than existing traditional data aggregation and authentication algorithms.

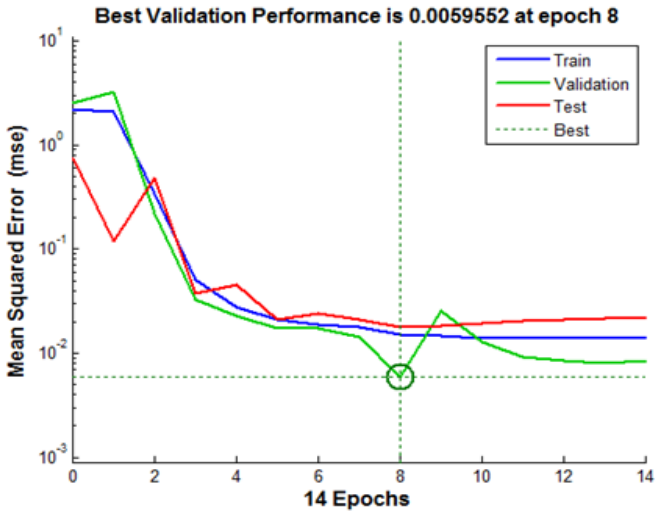

Figure 9. A Performance Plot

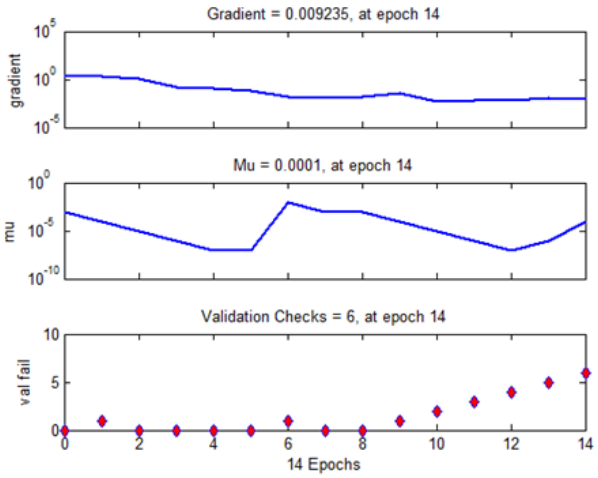

Figure 10. The Training State

5. Conclusions

This paper proposes a combined system to resolve some longstanding issues in the WSN technology. The similarity between WSN and neural networks suggests that it may be

reasonable to combine these technologies. In particular, correct information and data aggregation are crucial in WSNs because sending incorrect data by fault sensors make wrong decisions on sensor environments and an increase in defective sensors. And these incorrect data decries reliability of WSNs. This paper considers a case in which an artificial neural network is securely performed over a WSN. To do this, Holenderski *et al.*, decomposition model was revised to support secure computing. The results show that such a combined system can indeed be constructed and that its performance is similar to that of data aggregation and authentication algorithms. Because of the distinct characteristics of WSNs, public keys do not need to be authenticated in the same way as it is done in the internet environment. Instead, public keys can be authenticated using the SHA-1 algorithm, which is much more efficient than the signature verification of certificates and thus can increase the efficiency of data transfer by reducing the loss and delay of data.

References

[1] M. Lotfinezhad and B. Liang, "Effect of partially correlated data on clustering in wireless sensor networks", Proceedings of the IEEE International Conference on Sensor and Ad hoc Communications and Networks (SECON), **(2004)**; Santa Clara, California.
[2] S. Soro and W. Heinzelman, "Prolonging the Lifetime of Wireless Sensor Networks via Unequal Clustering", Proceedings of the 5th International Workshop on Algorithms for Wireless, Mobile, Ad Hoc and Sensor Networks (IEEE WMAN '05), **(2005)**.
[3] K. Kalpakis, K. Dasgupta and P. Namjoshi, "Maximum Lifetime Data Gathering and Aggregation in Wireless Sensor Networks", Proceedings of IEEE Networks, **(2002)**.
[4] A. Kaur, V. Kaur and A. Pathak, "Data Aggregation Approach Using Neural Network in Wireless Sensor Networks", International Journal of Technological Exploration and Learning (IJTEL), vol. 1, no. 1, **(2012)**, pp. 5-9.
[5] S. Ozdemir and Y. Xiao, "Secure data aggregation in wireless sensor networks", A comprehensive overview, Computer Networks, vol. 53, **(2009)**, pp. 2022–2037.
[6] M. Cardei and D. Du, "Improving wireless sensor network lifetime through power aware organization", ACM Journal of Wireless Networks, vol. 11, no. 3, **(2005)**, pp. 333-340.
[7] R. Shah and J. Shah and J. Rabaey, "Energy Aware Routing for Low Energy Ad Hoc Sensor Networks, in Rabaey, Energy Aware Routing for Low Energy Ad Hoc Sensor Networks", Proceedings of the IEEE Wireless Communications and Networking Conference (WCNC), **(2002)**; Orlando, Florida.
[8] B. Saeid, K. Helia and D. Ladan, "Neural networks for error detection and data aggregation in wireless sensor network", International Journal of Computer Science Issues, vol. 8, no. 5, **(2011)**, pp. 287-293.
[9] S-H. Lee, S-J Lee and K-I Moon, C"Life-Environmental Sensor Data Aggregation Based on Neural Network", Advanced Science and Technology Letters, vol. 47, **(2014)**, pp. 204-207.
[10] M. Holenderski, J. Lukkien, and T. C. Khong, "Trade-offs in the Distribution of Neural Networks in a Wireless Sensor Networks", First International Workshop on Data Mining in Sensor Networks, **(2005)**.
[11] E. Fasolo, M. Rossi, J. Widmer, and M. Zorzi, "In-Network Aggregation Techniques for Wireless Sensor Networks: A Survey", IEEE Wireless communication, vol. 14, no. 2, **(2007)**, pp. 70-87.

Authors

Sang- Hyun Lee, he received the BS and MS in Department of Computer Engineering from Honam University. in 2002 and 2004, respectively. He received Ph.D. degrees in Computer Science from Chonnam National University in 2009. He has been a professor at Honam University since 2012. His research interests include artificial intelligence, Software Engineering, Early Warning System, claim analysis, intelligence automotive

Sang-Joon Lee, he received the BS, MS and Ph.D. degrees in Computer Science and Statistics from Chonnam National University. He was assistant professor in Seonam University and Shingyeong University. Since 2007, He has been with Chonnam National Univ. as an associate professor in the School of Business Administration. His research interests include MIS, SE, IT Service and Ubiquitous Business.

Kyung-li Moon, he is a professor at the Department of Computer Engineering, Honam University in Gwang-Ju, Korea. His theoretical work began at Seoul University as a statistical computing scientist, and then expanded into complexity science, chaos theory, and cognitive science generative sciences.