Scientific Research Publishing

Journal of Computer and Communications, 2017, 5, 10-23
http://www.scirp.org/journal/jcc
ISSN Online: 2327-5227
ISSN Print: 2327-5219

A Model of Security Adaptation for Limited Resources in Wireless Sensor Network

Jumadi Mabe Parenreng[1,2], Akio Kitagawa[1]

[1]Electrical Engineering and Computer Science, Kanazawa University, Ishikawa, Japan
[2]Information Engineering and Computer Education, State University of Makassar, South Sulawesi, Indonesia
Email: parenreng@merl.jp, kitagawa@is.t.kanazawa-u.ac.jp

How to cite this paper: Parenreng, J.M. and Kitagawa, A. (2017) A Model of Security Adaptation for Limited Resources in Wireless Sensor Network. *Journal of Computer and Communications*, **5**, 10-23.
https://doi.org/10.4236/jcc.2017.53002

Received: January 6, 2017
Accepted: March 10, 2017
Published: March 13, 2017

Abstract

View of wireless sensor network (WSN) devices is small but have exceptional functionality. Each node of a WSN must have the ability to compute and process data and to transmit and receive data. However, WSN nodes have limited resources in terms of battery capacity, CPU, memory, bandwidth, and data security. Memory limitations mean that WSN devices cannot store a lot of information, while CPU limitations make them operate slowly and limited battery capacity makes them operate for shorter periods of time. Moreover, the data gathered and processed by the network face real security threats. This article presents an Adaptable Resource and Security Framework (ARSy) that is able to adapt to the workload, security requirements, and available resources in a wireless sensor network. The workload adaptation is intended to preserve the resource availability of the WSN, while the security adaptation balances the level of security with the resource utilization. This solution makes resources available on the basis of the workload of the system and adjusts the level of security for resource savings and makes the WSN devices work more efficiently.

Keywords

Wireless Sensor Network, Resource-Aware, Security-Aware, Workload System, Limited Resources

1. Introduction

Wireless sensor network (WSN) technology enables continuous monitoring, tracking, and control. WSN technology is the basis for acquiring information with which to make decisions and take actions on, for example, weather forecasts, traffic information, currency movements, and public transportation schedules. It can even be used to monitor states and places that are inaccessible to

humans, such as the movements of tectonic plates deep underground, volcanic activity, and radiation levels in areas affected by nuclear accidents.

WSN devices must be small and have exceptional functionality. They must have the ability to receive data, process the data, and send the results. To monitor a large area, WSN devices are supported by the network and other devices throughout it. To avoid unnecessary detections and storage of data, a filtering mechanism using data mining algorithms should be implemented in each node [1] [2] so that only the most important data is saved [3].

WSN devices have limited resources, such as battery capacity, memory, CPU, and radio communication capacity [4] [5] [6]. Not all WSN devices have sufficient data processing or storage capabilities, and few have good batteries. In addition, security threats are a major concern; data obtained by a WSN may be of little use if there is no guarantee of its security. The security-aware concept described in this study gives a WSN the ability to adapt the level of security to the workload [7] [8] [9] and balance resource utilization and the level of data security [7] [10] [11].

This article presents a framework that is able to adapt the availability of resources and level of security to the workload on the wireless sensor network system. The rest of the paper is outlined as follows. Section 2 discusses the literature related to this topic, Section 3 discusses the ideas proposed in this study, Section 4 discusses the evaluation of the offered solutions, and Section 5 concludes the paper and discusses future work.

2. Related Work

2.1. Resource of WSN Node

Resource availability is a very important consideration for a WSN. Jason [12] states that the resources of a WSN consist of a battery, radio, processor, and sensor. The author goes on to describe how resources can be combined to make possible thousands of applications for the public good. Karl *et al.* [6] describe WSN resources as consisting of a controller, memory, sensors, actuators, communication capacity, and a power supply. In particular, the battery or power supply is necessary for the other WSN equipment to operate so that their availability can be maintained [13].

To maintain the availability of a resource, researchers have tried to modify the algorithm in use. In particular, Gaber *et al.* [1] discuss how the battery, CPU, and memory can be utilized in ways that can increase the lifetime of the network. They use a resource monitoring scheme to track resources of nodes. Their scheme works by monitoring the conditions of availability of the main resources. Significant changes to the availability affect the adaptation performance [5] [14].

A gradually changing resource availability is a challenge, because the availability of the battery resource especially determines whether or not the system continues operation [13]. WSN states such as active, idle, and off, consume battery power. In particular, the idle state has been found to consume10 mW [15], while the off state consumes 0.016 mW [16].

Gaber and Yu [2] describe algorithm granularity, that is, putting settings on input by using an input algorithm, putting settings on processes by using a process algorithm and putting settings on output of the algorithms, all with the aim to save battery, CPU, and memory [5] [17] [18] [14]. **Figure 1** shows the adaptation mechanism.

2.2. Resource Awareness

The input algorithm [1] controls the availability of battery resources. Adaptation is done in sampling intervals (SI) [17]. In previous research [18], if the battery capacity is 100%, each data stream is checked, but if the battery capacity drops to a critical level, then data is checked only in certain intervals decided on the basis of the availability of the battery resource, which is calculated using the sampling interval formula. The sampling interval (*SI*) formula is as follows [17]:

$$SI = ub - \text{battery}\frac{ub - lb}{batt\_crit\_\text{threshold}} \tag{2.1}$$

Here, *ub* is an upper bound for the maximum availability of a battery resource, *lb* is a lower bound for the minimum availability of a battery resource, *battery* is the availability of the battery capacity at the time, and *batt_crit_threshold* is the threshold availability of a battery in a critical condition.

Process Algorithm [1] controls the availability of the processor resources (CPU). Adaptation is done through the random factor (*RF*) calculation below [17]. Here, if the CPU capacity is 100%, then each data stream is checked to determine the proximity of the data. If the CPU resource has reached a critical level, the data streams are checked randomly [18].

$$RF = \frac{10.000 - cpu\_crit\_\text{threshold} * lb - (100 - lb) * cpu}{100 - cpu\_crit\_\text{threshold}} \tag{2.2}$$

Here, *cpu_crit_threshold* is the critical threshold of the CPU, *cpu* is the CPU resource capacity at the time, and 100 is the maximum utility of the CPU.

The output algorithm [1] controls the availability of the memory resources. The adaptation is done through the radius threshold (RT) calculation below [17]; if the available memory drops below the critical limit, then the system will reduce the usage of memory by limiting the number of clusters or counter formed, in order to free up memory space [18].

$$\text{radius} = ub - \text{memory}\frac{ub - lb}{mem\_crit\_\text{threshold}} \tag{2.3}$$

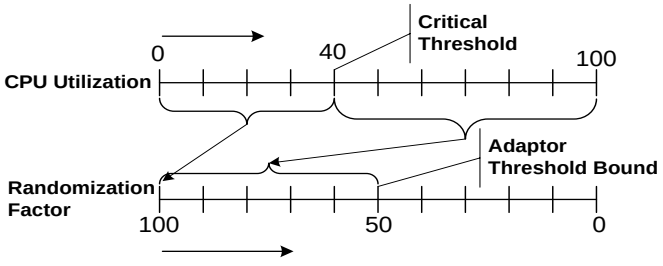

Figure 1. Adaptation mechanism [4].

Here, *radius* is a radius threshold that is the maximum distance threshold, memory is the total memory used at that time, and *mem_crit_threshold* is the critical limit of memory.

The monitoring system described above is updated by the resource monitor [1] [4] so that the processes that occur are always based on the current state of the device resources.

2.3. Mining Data

Mining data with WSN technology is still a relatively new idea. Here, data mining is performed on data sources in real time, so that the results of a process will have an impact on decision-making. This is a very challenging task for researchers because WSNs have limited computation and radio communication resources, and the volume of data generated by the WSN varies [19]. Another challenge is data mining of high-speed data streams [1] [2] [3] [17].

Researchers previously tried mining data sources using WSN devices, to find the latest information in a collection of data in real time. They did so by using data analysis techniques to extract useful information for the end user [20]. There are several ways of mining data, such as by using clustering algorithms [1] [21], classification algorithms [17] [22], frequent pattern algorithms [18] [23], and association rules algorithms [24].

2.4. Resource-Aware and Mining Data Algorithm

2.4.1. Resource-Aware Cluster Algorithm

The resource-aware clustering algorithm (RA-Cluster), introduced by Gaber *et al.* [2], has two main components. The resource aware (RA) component uses adaptation techniques to obtain data on high-speed data streams with maximum accuracy based on the availability of data. The algorithm begins by examining the minimum data rate. If the algorithm can operate at the minimum data rate, the RA component tries to find a solution that is able to maximize accuracy through the increased data rate. Otherwise the algorithm sends requests to the data mining server in order to achieve minimum accuracy [25]. The second component is a clustering algorithm, whose steps are as follows [2] [3] [25],

a) The data items arrive in the order of the data rate.
b) The algorithm determines a starting point as a data center.
c) It compares this center with each new data item with a central point to get the distance to that data.
d) If the distance to all centers is greater than the threshold, the new items become a new center. If not, the weights for the data center point to the closest data item are raised, and the new center is equal to the weighted average.
e) Repeat step c and d,
f) If value centers = k (based on available memory), create a new centers vector,
g) Repeat step c, d, e and f,
h) If memory is full, then re-cluster.

2.4.2. Resource-Aware Classification Algorithm

The resource-aware classification algorithm (RA-Class) was also introduced by Gaber *et al.* [2] [17] [22]. Initially, the algorithm determines the number of instances based on the availability of memory. When the element class of the data arrives, the algorithm seeks a nearby instance that has been in the memory, on the basis of a certain threshold distance. The algorithm uses this threshold to determine whether two or more elements are similar. If so, the algorithm checks the class labels. If the class labels are the same, the weights are increased by 1. If not, the weights are reduced by one. If the weight decreases to zero, the corresponding element is removed from memory. This algorithm changes the distance threshold from time to time in order to cope with the speed of the incoming data.

2.4.3. Resource-Aware Frequent Item Algorithm

The resource aware frequent item algorithm (RA-Frequent Item) calculates the number of frequent data items on the basis of the availability of memory [3]. This value is continuously updated to deal with high data rates. The algorithm represents the number of frequent items as a counter that is reset after the time limit is reached, to cope with the changing nature of the data stream. The algorithm receives data elements one by one and tries to find a counter for each new data, and if it succeeds, it increases the number of items for the same data. If all counters have been filled, some of the new items are ignored and the counter is reduced by one, until the algorithm reaches the specified time limit. A counter with the least frequent data is ignored, and the counter is reset to zero. If a new item is the same as a data item in memory based on the similarity threshold, the average value of both items is allocated and the counter is incremented by one. The main parameters including time threshold and number of counter items affect the accuracy of the algorithm. When the threshold time is reached, the algorithm deletes the remaining items and resets the counters [2] [18] [23] [26].

2.5. Security Awareness

In the past, data models were manually developed at great expense; it took a long time, requiring hardcopy documentation and a large physical storage space. Today, WSN devices are able to create more detailed models faster and more efficiently that can be updated at any time.

Maintenance and control devices are highly dependent on the placement and function of the WSN. Sensors may be placed in areas that are too humid, dusty, or hot to do maintenance. WSNs can economically monitor extreme environments that are inaccessible to humans [27]. Despite their extraordinary abilities, WSNs have disadvantages, *i.e.*, limited resources, unreliable communications, and unattended operation [8]. In particular, the data exchange mechanism, which uses wireless media communication, is vulnerable to valid and invalidated data packets, data conflicts, and errors in data transfer. Latency may also trigger failures when synchronization between sensor nodes is important.

Bogdan *et al.* [10] describes how to assign a protection level to a transaction and assign a proper security level on transaction data to improve system performance.

2.6. Security Level

Security adaptation based on resource availability is very helpful to optimize security [10]. Efficient resource usage has a direct impact on operation of WSN devices. The higher the security level, the greater the cost and impact on processing time resources [7].

T. Xie *et al.* [28] assign security levels (0.1-0.9) based on a hash function for integrity. Bogdan *et al.* [11] assign five levels of security: very low, low, medium, high, and very high. The challenge for them was optimization of security in mobile devices.

2.7. Workload

Discussions regarding workload always involve optimization and efficiency paired with the concept of scheduling. Bo Zeng *et al.* [29] schedule transmissions of data packets on each node on the basis of the workload node concept. Another study that uses the concept of the workload node is reported by Fan Wu *et al.* [30], which proposes workload-aware channel assignment (WACA) algorithm for lossy channels in urban networks. T. Xie *et al.* [28] discuss a workload-aware MAC protocol (W-MAC) for the heterogeneous environments of WSNs, where each sensor node generates data with a different capacity. The scheduling concept and workload-aware time slice allocation mechanism minimize the power consumption of the node, to cope with delay of the data and adjust the schedule to the variable data rate due to the changing network topology.

3. Adaptable Resource and Security (ARSy) Framework

The term ARSy Framework is an adaptable resource and security framework. The framework describes the relationship between blocks, as shown in **Figure 2**. The details of this process are explained below.

The ARSy Framework consists of three main blocks. The first is the Client Node, which is equipped with resources to perform data processing. The second block processes the data. The third block is called the Server Data block which is in different area from the node and this research is done until to the delivery of the results of the node as the final destination of all the data.

3.1. Data Input

The data input block was developed in previous research [18]. It receives the data streams to be processed. It executes a data mining process that determines which of the data are similar [1].

3.2. Resource Monitor

The resource monitoring block keeps track of the availability of resources such

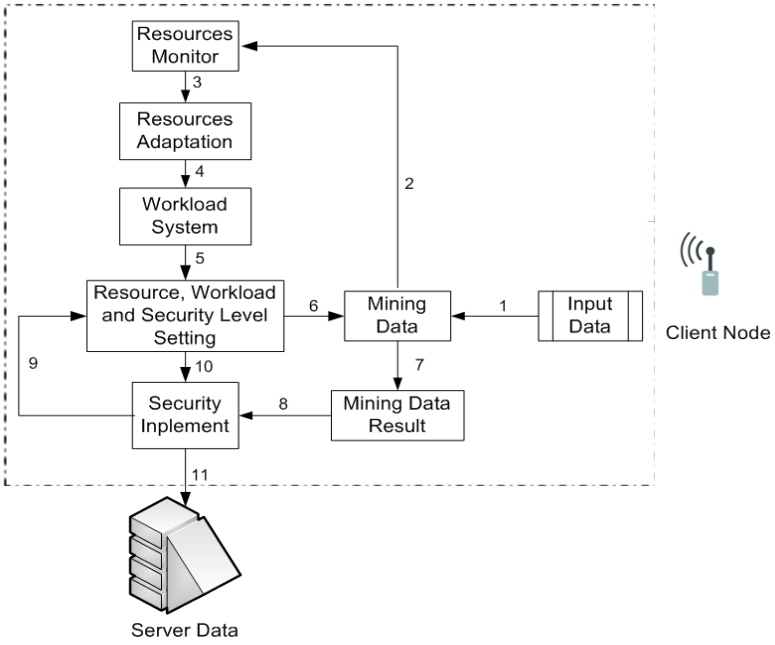

Figure 2. ARSy framework.

as memory, CPU, and battery. There are two alternative models. The first reports the resources to be used by the data mining block. The monitor does not report all resource changes that occur to the data mining block, only those in which resource availability decreases by more than 5% (>5%). If such a resource decrease occurs, it updates its information sent to the resource, workload, and security level settings block. Then the data mining block processes the data with the available resources. The process continues until the resource reaches a critical threshold, at which time the adaptation system react, for e.g., battery via IS, CPU via RF and memory via RT.

In the second model, changes in a resource are not addressed by the system resource adaptation until the resource is in a critical condition. The resource monitoring block sends an update on resource availability to the resource, workload and security level settings block when resource of nodes are in a critical condition. This adaptation applies equally to all types of resources until a resource is completely exhausted and the system no longer functions.

3.3. Resources Adaptation

3.3.1. Battery

In battery adaptation through the sampling interval (SI), stream data is continuously processed. If the battery is the maximum resource, each data is processed to determine its type, similarity with previous data, and new counters are created for each piece of new data is not similar to other data. When the battery capacity falls below a certain threshold, data is collected at a specified interval based on the amount of resource remaining. The degradation of resource availability affects the interval and data processing until the battery completely runs out.

Figure 3 shows the battery adaptation algorithm is described as follows. The system accepts an input data stream, if the resource battery availability is greater than the value of the sampling interval and the sampling interval of the battery is equal to the lowest threshold adaptation of the battery, this condition is ideal, so that the system does not implement adaptation policies. System update its latest battery setting, processing of all data passing and stores the results of the data mining process. Other conditions when the battery resource falls below the minimum value for the sampling interval, the value of sampling interval is re-calculated, update on the latest battery resource value to the resource monitoring block then implement values adaptation and stores the results of the data mining process.

3.3.2. CPU

CPU adaptation through the Random Factor (RF). If the CPU resource availability is above the critical threshold, all data passed to the CPU is processed. When the resource falls below a critical threshold, the data to be processed is drawn randomly.

Figure 4 shows the CPU adaptation algorithm is describe as follows. The system accept an input data stream, if the resource CPU availability is greater than the value of the random factor CPU, its mean the threshold of the CPU (RF_CPU) is 100%, system update its latest CPU setting and all data are randomized to determine their similarity to existing counters, then stores the result of the data mining process. If the resource availability falls below the critical threshold of RF_CPU, RF_CPU is recalculated and used as a reference value for CPU adaptation policies.

3.3.3. Memory

Memory adaptation using the radius threshold (RT) is performed on the basis of proximity data. The results of the data mining process in a specific time period are stored in memory. **Figure 5** shows the pseudocode for memory adaptation.

```
Algorithm 1: Battery Adaptation Policy
//Resource Battery Adaptation
//Assumption Battery capacity 100%
START
Input data stream
IF(Available BAT > SI_BAT)
        SI_BAT = SI_Lower_Bound
        Setting Battery resource condition monitored
        Process all data stream
        Save mining result to maximum counter
ELSE
        Calculate SI_BAT
        Setting Battery resource condition monitored
        Implement Battery adaptation
        Save mining result to minimum counter
END IF
STOP
```

Figure 3. Pseudocode for battery adaptation.

```
Algorithm 2: CPU Adaptation Policy
//Resource CPU Adaptation
//Assumption CPU Capacity 100%
Input data stream
IF(Available CPU > RF_CPU)
        RF_CPU = 100%
        Setting CPU resource condition monitored
        Randomize all data stream
        Save mining result to maximum counter
ELSE
        Calculate RF_CPU
        Setting CPU resource condition monitored
        Implement CPU adaptation
        Save mining result to minimum counter
END IF
STOP
```

Figure 4. Pseudocode for CPU adaptation.

```
Algorithm 3: Memory Adaptation Policy
//Resource Memory Adaptation
//Assumption Memory capacity 100%
START
Input data stream
IF(Available MEM > RT_MEM)
        RT_MEM = RT_Lower_Bound
        Setting Memory resource condition monitored
        Save mining result to maximum counter
ELSE
        Calculate RT_MEM
        Setting CPU resource condition monitored
        Implement Memory adaptation
        Save mining result to minimum counter
END IF
STOP
```

Figure 5. Pseudocode for memory adaptation.

Memory adaptation does not occur if the available memory is greater than the radius threshold. All the data mining results are stored. When the available memory capacity falls below the radius threshold, the radius threshold is recalculated, the resource monitor updates its resource memory information and the data mining results are sent to a minimum counter.

The radius threshold grows and shrinks according to resource availability. If the available memory capacity is larger, more counters of data mining results are stored in a data mining period. But if the memory capacity falls below the threshold, only the minimum number of counters is stored in a data mining period.

3.4. Workload System

Nodes could be placed in extreme environments and have heavy workloads. The conditions in which the device has to operate is an important consideration. Here, a heavy workload condition means that one of the resources such as battery, CPU or memory has reached a critical condition, and adaptation occurs. A

light workload means that the resources are in a normal condition or at maximum, and there is no resource adaptation, the algorithm shows in **Figure 7**.

3.5. Resource, Workload and Security Level Setting

Battery and memory on a resource node will decrease over a certain period of time. Information on resource availability during a specific time period is a reference for the data mining process to decide amount of resource that it uses. The resource availability information is updated continuously.

There are three setting operations. The first is the resource settings for memory, CPU and battery, which uses information sent by the resource monitor. These settings are used as input parameters for the data mining process. The second is the workload setting determining a heavy workload or a light workload. The third is the security setting, which determines the level of security to be applied to each of the data mining output based on the conditions of the resources and workload at the time.

3.6. Data Mining and Output

Figure 6 shows the lightweight frequent item (LWF) algorithm [18] [23] operates on the received data streams and considers the availability of battery, CPU, and memory. The results are placed in a counter determined by the category of the data and are limited to a specific time period. When the time limit expires, only a counter that has the highest data frequency is used to store data. The results of this process are the final output.

3.7. Security Level

Ideally, data has the maximum security level. The higher security level is, the greater the resources are needed [7] to maintain it, and thus, limited resource availability means that such a level cannot always be maintained in WSNs [28]. The security level is also representative of the amount of resources required by the device. The security levels are high, medium, low, and very low.

```
Algorithm 4: Light Weight Frequent Item (LWF Algorithm)
//Algorithm for Mining Data Process
Set the number of top -n frequent items to K.
Set a counter for each K.
Repeat
        a.  Receive the item
        b.  If the item is new and one of the K counters is 0
            Then
                Put this data category and increase the counter by 1.
            Else
                If the item is already in one of the K counters
            Then
                Increase the counter by 1.
            Else
                If the item is new and all the counters are full
            Then
                Check the time
                If the time > Threshold Time
                Then
                Re-set the number of least -n K counters to 0
                Put this data the new item and increase the counters by 1.
            Else
                Ignore the item
            Decrease all the counters by 1.
Until Done
STOP
```

Figure 6. Lightweight frequent item algorithm [2].

4. Solution and Evaluation

4.1. Workload Solution

The workload of a WSN may vary over time. When the WSN is in an overloaded condition, the condition may resolve itself or ends with a deadlock.

Workload detection is done by checking the state of the battery, CPU, and memory. If the memory resource is above the average level of 50%, the condition of the resource is considered a maximum, so the workload is considered to be light and no resource adaptation occurs. When the resource falls below 50%, the workload is heavy and an adaptation policy is applied to the critical resource. The system updates the workload conditions that affect the level of security of the data before it is sent to the server. **Figure 7** shows the pseudocode for workload detection.

To determine workload conditions, the resources all have a maximum value of 100%. The average total value of battery, CPU and memory is Σ〖Resource = Battery + CPU + Memory/3〗. If one of the resources is in a critical condition, but the average resource is greater than 50%, the node status is still considered to be a light workload. However, if the resource average is less than 50%, the node is considered to have a heavy workload and resource adaptation is implemented.

4.2. Security Level Solution

Table 1 lists adaptations based on resources, workload and level of security.

```
Algorithm 5: Workload Detection
//Algorithm for Workload Detection WSN
//Workload status LIGHT or HEAVY
START
Check available Resource (BAT, CPU, MEM)
IF(Resource > Threshold BAT, CPU, MEM)
        System LIGHT workload
        System without adaptable
        Update workload setting
ELSE(Resource < Threshold BAT,CPU,MEM)
        System HEAVY workload
        Implement policy adaptable resource
        Update workload setting
END IF
STOP
```

Figure 7. Pseudocode for workload detection.

Table 1. Relationship between resource, workload and security level.

No	Resource	Workload	Security Level	Average Resource (%)
1	Maximum	Light	High	100-75
2	Maximum	Heavy	Medium	75-50
3	Minimum	Light	Low	50-20
4	Minimum	Heavy	Very Low	20-0

In the first condition, the resource is maximum and workload is light, and the system applies a high-level security policy, assuming the average resource availability equal to 75% - 100%. In the second condition, the resource is maximum and workload heavy, and the system applies a medium level security policy, assuming the average resource availability equal to 50% - 75%. In the third condition, the resource is minimum and workload light, and the system implements a low-level security policy, assuming the average resource availability equal to 20% - 50%. In the fourth condition, the resource is minimum and workload heavy, and the system applies a very low level security policy, assuming an average resource availability equal to 0% - 20%. **Figure 8** shows the pseudocode for security level adaptation.

The security level is the last adjustment. The system receives information on resource availability and workload conditions from the resource, workload and security level block. If the average resource is greater than the threshold, the resource condition is considered to be maximum. If there was no resource adaptation, the workload is considered light, so the security policy applied to the data is high level. If a heavy workload is detected, a medium security level is implemented. When the resource is minimum, which means that the average availability of all the resources is below the threshold of the particular resource at that time when no adaptation has been implemented, the workload is categorized as light and the level of security is low. When all the resources are minimum and the workload heavy, adaptation occurs and the security level is low. This condition continues until the WSN resources run out.

```
Algorithm 6: Security Level Implementation
//Algorithm for Security Level of Data Output
//Security Level Very Low, Low, Medium, and High
START
Input resource info
Input workload info
IF(Average Resource > Threshold)
        Resource MAXIMUM condition
        IF(!Resource Adaptation)
                Workload LIGHT
                Implement security level HIGH
        ELSE
                Workload HEAVY
                Implement security level MEDIUM
        END IF
ELSE
        Resource MINIMUM condition
        IF(!Resource Adaptation)
                Workload LIGHT
                Implement security level LOW
        ELSE
                Workload HEAVY
                Implement security level VERY LOW
        END IF
END IF
STOP
```

Figure 8. Pseudocode for security level adaptation.

5. Conclusion

Resource saving is very important when a WSN has limited resource availability and is deployed in extreme environments without any chance for maintenance. In addition, while maximizing data security is a good idea, the level of security should be determined in a way that considers the limited resources of the WSN, so that it can survive for long period of time. A higher security level imposes a greater cost on and shortens the lifetime of the WSN devices. This study describes a resource availability adaptation for WSNs that is based on the workload of the system and adjusts the level of the security for resource savings.

Acknowledgements

Jumadi Mabe Parenreng is supported by special fund for the Doctoral Program, by Ministry of Research, Technology and Higher Education of the Republic of Indonesia, Guarantee Letter No. 985/E4.4/K/2015 and Akio Kitagawa is supported by JSPS KAKENHI Grant Number 25286036, 15K12504, and CREST, Japan Science and Technology Agency.

References

[1] Gaber, M.M. and Yu, P.S. (2006) A Framework for Resource-Aware Knowledge Discovery in Data Streams: A Holistic Approach with Its Application to Clustering. *Proc.* 2006 *ACM Symp. Appl. Comput - SAC*'06, 649-656. https://doi.org/10.1145/1141277.1141427

[2] Gaber, M.M., Krishnaswamy, S. and Zaslavsky, A. (2003) Adaptive Mining Techniques for Data Streams Using Algorithm Output Granularity. *AusDM*.

[3] Gaber, M.M., Krishnaswamy, S. and Zaslavsky, A. (2004) Cost-Efficient Mining Techniques for Data Streams. *Proc. Australas. Work. Data Min. Web Intell.* (*Dmwi*2004), **32**, 109-114.

[4] Phung, N.D., Gaber, M.M. and Roehm, U. (2007) Resource-Aware Distributed Online Data Mining for Wireless Sensor Networks. *Int. Work. Knowl. Discov. From Ubiquitous Data Streams*, 59.

[5] Shiddiqi, A.M. Performance Measurement of Resource-aware Framework in Online Data Stream Mining, 1-7.

[6] Karl, H. and Willig, A. (2005) Protocols and Architectures for Wireless Sensor and Architectures for Wireless Sensor.

[7] Son, S.H., Zimmerman, R. and Hansson, J. (2000) An Adaptable Security Manager for Real-Time Transactions. *Proc. - Euromicro Conf. Real-Time Syst.*, 63-70. https://doi.org/10.1109/emrts.2000.853993

[8] Walters, J. and Liang, Z. (2006) Wireless Sensor Network Security: A Survey. *Secur. Distrib. Grid, Pervasive Comput.,* 368-404.

[9] Zhu, X., Guo, H., Liang, S. and Yang, X. (2012) An Improved Security-Aware Packet Scheduling Algorithm in Real-Time Wireless Networks. *Information Processing Letters*, **112**, 282-288. https://doi.org/10.1016/j.ipl.2011.11.018

[10] Ksiezopolski, B. and Kotulski, Z. (2007) Adaptable Security Mechanism for Dynamic Environments. *Computer Security*, **26**, 246-255. https://doi.org/10.1016/j.cose.2006.11.002

[11] Ksiezopolski, B., Szalachowski, P. and Kotulski, Z. (2010) SPOT: Optimization Tool

for Network Adaptable Security. *Communications in Computer and Information Science*, **79**, 269-279. https://doi.org/10.1007/978-3-642-13861-4_28

[12] Hill, J.L. (2003) System Architecture for Wireless Sensor Networks. Spring, 186.

[13] Penella, M.T., Albesa, J. and Gasulla, M. (2009) Powering Wireless Sensor Nodes: Primary Batteries versus Energy Harvesting. 2009 *IEEE Intrumentation Meas. Technol. Conf. I2MTC* 2009, May 2009, 1625-1630.

[14] Parenreng, J.M., Syarif, M.I., Djanali, S. and Shiddiqi, A.M. (2011) Performance Analysis of Resource-Aware Framework Classification, Clustering and Frequent Items in Wireless Sensor Networks. *Proceeding Int. Conf. eEducation Entertain. eManagement*, 117-120.

[15] Shih, E., Cho, S.-H., Ickes, N., Min, R., Sinha, A., Wang, A. and Chandrakasan, A. (2001) Physical Layer Driven Protocol and Algorithm Design for Energy-Efficient Wireless Sensor Networks. *Proc. 7th Annu. Int. Conf. Mob. Comput. Netw. - MobiCom* '01, 272-287. https://doi.org/10.1145/381677.381703

[16] Schurgers, C., Tsiatsis, V., Ganeriwal, S. and Srivastava, M. (2002) Optimizing Sensor Networks in the Energy-Latency-Density Space. *{IEEE} Trans. Mob. Comput.*, **1**, 70-80.

[17] Shiddiqi, A.M. and Gaber, M.M. Light Weight Resource-Aware Data Stream Classification, 1-6.

[18] Parenreng, J.M., Djanali, S. and Shiddiqi, A.M. (2010) Analisa Kinerja Resource-Aware Framework Pada Algoritma Light-Weight Frequent Item (LWF). *Proceeding SNPI ITS Surabaya*.

[19] Mahmood, A., Shi, K., Khatoon, S. and Xiao, M. (2013) Data Mining Techniques for Wireless Sensor Networks: A Survey.

[20] Mahmood A., Shi, K. and Khatoon, S. (2012) Mining Data Generated by Sensor Networks A Survey. *Information Technology Journal*.

[21] Aggarwal, C.C. and Wang, J. Chapter 2: On Clustering Massive DATA Streams : A Summarization Paradigm.

[22] Shiddiqi, A.M. (2009) Resource-Aware Data Stream Classification in Wireless Sensor Network. *Thesis Master*, 1–72.

[23] Orlando, F.S.S., Palmerini, P. and Perego, R. Adaptive and Resource-Aware Mining of Frequent Sets.

[24] Agrawal, R. and Srikant, R. (2008) Fast Algorithms for Mining Association Rules. *Ann. Pharmacother.*, **42**, 62-70.

[25] Gaber, M.M., Krishnaswamy, S. and Zaslavsky, A. A Wireless Data Stream Mining Model. *Architecture*.

[26] Chang, Y.-I., Li, C.-E. and Peng, W.-H. (2012) An Efficient Subset-Lattice Algorithm for Mining Closed Frequent Itemsets in Data Streams. 2012 *Conf. Technol. Appl. Artif. Intell.*, 21-26.

[27] Akyildiz, I.F., Su, W.L., Sankarasubramaniam, Y. and Cayirci, E. (2002) A Survey on Sensor Networks. *IEEE Communication Magazine*, 102-114.

[28] Xie, T., Qin, X. and Sung, A. (2005) SAREC : A Security-Aware Scheduling Strategy for Real-Time Applications on Clusters.

[29] Zeng, B., Dong, Y., Liu, Z. and Lu, D. (2012) A Workload-Aware Link Scheduling for Heterogeneous Wireless Sensor Networks. 2012 *Int. Conf. Cyber-Enabled Distrib. Comput. Knowl. Discov.*, 353-359.

[30] Wu, F. (2012) Workload-Aware Opportunistic Routing in Multi-Channel, Multi-Radio Wireless Mesh Networks, 344-352.