A Study of Limited Resources and Security Adaptation for Extreme Area in Wireless Sensor Networks

Jumadi Mabe Parenreng*, Akio Kitagawa
Electrical Engineering and Computer Science, Kanazawa University, Japan

Dyah Darma Andayani
Informatics and Computer Engineering Education, Universitas Negeri Makassar, South Sulawesi, Indonesia

*Corresponding author: jparenreng@unm.ac.id

Abstract. WSNs have five main components namely, the sensing device, processor, memory, power supply, and transceiver. The power supply, processor, and memory are the main resources and have limited resources; therefore, resource availability in WSNs must be maintained. With limited resources available, the WSNs is required to be able to work as efficiently as possible, operated in a long time period and secure due to its placement in extreme areas. Another challenge is to choose the WSNs that has short time operation with strong security or long time operation with adaptable security. This article provides limited resource solutions to WSNs whose placement is in extreme areas that are impossible to do maintenances. As a solution, an adaptation approach to resource availability and security is used as offered by the ARSy Framework. For testing, we use components such as Raspberry pi 3 Model B and DS18B20 temperature sensor. The advantage the raspberry pi because its CPU and Memory resources have a large capacity. With these advantages, it is highly manageable, allows to integrate several types of sensors in one raspberry pi unit, and the use of battery resource becomes optional. The battery will be only used based on the design requirements because the battery consumption is wasteful. The result of the research shows performance ARSy framework compared to the system that works without ARSy framework mechanism.

Keywords: extreme area, WSN, ARSy framework, resource, security adaptation

1. Introduction

Wireless sensor network technology (WSNs) is known as a low-cost, small, applicable, very powerful and useful technology for a wide range of applications. It also enables to monitor and control the physical environment from remote locations with high accuracy and can be applied to various domains such us monitoring environment and agriculture, healthcare, public safety and military system, industry, and transportation system [1]. Usability of WSNs is not only for regular area monitoring, but it is also much more advanced and developed to monitor more difficult and even extreme areas that cannot be reached by humans. For extreme areas, it usually uses a new smart sensor that allows for areas such as underground, underwater, and space [2]. For areas that are difficult to reach usually this sensor network system are implemented without regular maintenance. Therefore, it required a mechanism to maximize the lifetime of the system. Lifetime sensor nodes rely heavily on the success
of system design whose determinant factors are based on several parts, such as fault tolerant, scalability, production cost, hardware constraints, network topology sensors, environment, transmission media and power consumption [3].

Average node sensor embedded resource that also adjusted with its small size makes WSNs equipped with limited resources. By limited resources, unreachable placement, and without alternative energy scavenging, then the only choice is by using a power battery. In this condition, the energy saving mechanism is required from all operational aspects of the sensor node. To overcome the problem of sensor node resources, it needs adaptation solution to resource availability through the efficient use of resource sensor node [4]. The solution is by reducing the amount of data communication by moving the data processing algorithm to the sensor node. Another solution is to combine data communications and data processing. To reduce the cost of data communications that occurred required data reduction techniques such as by implementing data mining algorithms on the sensor node. Then, for the efficiency of resource use, the adaptation is based on resource-level which gradually decreases, as does the availability of battery, CPU, and Memory [5][6][7].

Previous research proposes a method that sets up resource availability for streaming mining data, i.e., Algorithm Granularity Setting (AGS). The algorithm consists of three settings [8]. First, for Input Setting called Algorithm Input Granularity (AIG). This section sets the Input for the change in sampling rate or data structure of the data stream used by the stream mining algorithm in the data. Output Setting called Algorithm Output Granularity (AOG). This section performs settings for the size change output of the algorithm, for example, the number of clusters formed through the clustering algorithm. Third, Process setting called Algorithm Process Granularity (APG). This section sets the random factor change of algorithm to minimize the power consumption.

Another interesting issue is network security and data on sensor nodes. Traditionally, security offers a model of system protection as strong as possible, but most of the data protection levels are always higher than the required potential threats. When security policies are implemented very strongly, it will affect the overall performance of the system. Excessive protection will reduce reliability and availability and thus, will affect global security [9]. The appropriate level of security can be estimated in terms of providing protection model with different security quality [10]. Another threat when the system implements a very strong security policy that it can be a threat to the device performance that has limited resources. It also will be a way for new threats such as exhaustion of resources, whose impact will reduce system efficiency, availability and introduce redundancy. Another effect of excessive estimation of security will increase the complexity of the system, which then affects implementation. However, enforcing restrictions will reduce its function. As a solution, it can be done by predicting the appropriate level of security on each output data [11][7][4].

Adaptation of resources to maintain the availability of limited resources is always in a stable condition. Battery power will regularly decrease over time node operation. The busy CPU will affect the memory, and this activity will spend battery power faster than normal conditions so that the sensor activity will affect the lifetime sensor node. The availability of resources on the sensor node can be maintained by setting the input data mechanism through time capture. The CPU and Memory activities will be able to operate normally, and the battery resource will not decrease significantly. Security Adaptation is intended for data security, which does not require that node sensors encrypt with the highest security. The security adaptation to the sensor node resource is to encrypt data based on existing resources, so the most appropriate security level is a prediction mechanism for available resources on the sensor node is required.

Figure 1 below shows the relationship between Resource Aware and Security-Aware. One side of the sensor node resource must be maintained from time to time, while the security uses the resource to encrypt the data if it is driven with a strong security quality, then the limited sensor node resources will quickly run out, so the mechanism of security quality adjustment based on available resources.
This study implements a proposed adaptation model which is expected to be a solution to the problem of limited resources and security on the wireless sensor network. The model is build based on several requirements such as adaptable to resource condition and the workload, make efficient resource usage of WSN, improve the lifetime of WSN devices while operate under low resource, the increased lifetime sensor node which is able to operate under low resources, and create a security system that can adapt to the conditions of limited resources. Adaptation model is made through the security level; each level will be based on resource conditions that change from time to time.

2. Wireless Sensor Network (WSNs)

Resource-aware (RA) is an adaptation of a node sensor to resource conditions when one of its resources is critical. With RA implementation, critical resources can be recovered and will be in the position above the threshold value. For sensor nodes that use the battery as a power source, when the battery is critical, it will trigger the battery adaptation system. As the battery becomes the most important requirement, so the main purpose of battery adaptation is power conservation to extend lifetime [12][13][5]. Security Adaptation in WSN is sensitive issues [10]. In this is the case, this study intended for WSN node that has limited resources and placed on the extreme area. In extreme environments, wireless sensor systems (WSSs) use terms such as wireless sensors and network actuators, wireless smart, intelligent sensing, wirelessly connected distributed smart sensing, and unmanaged aerial vehicles (UAVs). The use of new technologies such as smart sensors will provide many possibilities for creating new technological systems and services [2].

2.1. Resource adaptation

The battery, CPU, and memory are the main resources; therefore, resource availability must be maintained. This is achieved through adaptation [14][4], whereas the parameters data and process are maintained based on our ARSy framework. The critical threshold limits each resource in a security system. If the threshold is exceeded, adaptation will be a trigger to reduce excessive resource usage.

| Resource | Definition | Parameters |
|----------|------------|------------|
| Battery  | $SI = ub - (bat_{available} \ast \frac{ub - lb}{batt_{crit\_threshold}})$ | SI: Sampling Interval, bat_{available}: free battery, ub: upper bound, lb: lower bound. |
| CPU      | $RF = (100 - CPU_{used}) \ast \frac{ub - lb}{100 - cpu_{crit\_threshold}}$ | RF: Random Factor, cpu_{crit\_threshold}: critical thres cpu, RT: Radius |
| Memory   | $RT = (100 - mem_{used}) \ast \frac{ub - lb}{100 - mem_{crit\_threshold}}$ | RT: Threshold, mem_{crit\_thres}: critical thres mem. |
When an adaptation occurs in one of the sensor-node resources by applying the specific adaptation policy to that resource (see Table 1). These mechanisms are described in more detail as follows [4]:

a. Input: Battery adaptation is a trigger on the input side and based on its resource available via the sampling interval (SI). If the usage exceeded the threshold, adaptation would be a trigger. Before battery-resource adaptation occurs, the input-data-collection time is normal (does not exceed the threshold), but adaptation is triggered when the input-data-collection time changes based on the available resources.

b. Process: CPU adaptation is triggered on the process-data side and is based on the availability of the processor resource (CPU) via the random factor (RF). If the CPU usage does not, exceed the threshold, all the collected data will be maintained on the counter data; otherwise, the system only stores some counter data with priority based on dominant and non-dominant counter data. Some non-dominant counter data will be eliminated to relieve the CPU; its value will be based on the RF value.

c. Output: Memory adaptation is triggered on the output-data side and based on the availability of the memory resource via the radius threshold (RT). When the resource memory is normal, the result is sent to the data server; as much as 50% of all counter data is saved. If the usage exceeds the threshold, the system will reduce memory utilization by limiting the amount of counter data created, and the data are stored as output data based on the RT presentation value.

2.2. Security adaptation

Security systems offer as much protection as possible; thus, power consumption will increase and the lifetime of the system will decrease. System services are reduced to decrease power consumption, which decreases the lifetime of the system [16]. In fact, security is usually higher than potential threats. When security is very strong, it affects the overall performance of the system such as will reduce reliability, availability and affect security globally. An appropriate level of security can be estimated in terms of providing different security-quality protection models for each type of data [17].

Previous studies focused on security policy to model the security level of data that may have different outputs generated over time because determining the security level of data based on the availability of resources. With a better availability of resources, the security level of data becomes higher [10][7].

The absolute requirement of a security system is the guarantee of high data security; however, in cases in which WSNs are used, high data security affects the performance and lifetime of the system because a higher data-security level means greater energy consumption for cryptographic data functions [18][19][20]. The solution is to balance the use of resources through the security level of data [10], which is basically used to offset the use of resources when their availability has entered a critical phase. The policy of applying a high level of security to each output affects the lifetime of a WSN because higher security levels of data put greater demands on the CPU and increase battery consumption [19].

The security-adaptation model applied in this study is for estimating the security level of output data based on the resource condition. When the resource condition does not exceed the threshold, the output data have the maximum security level, but when resource availability falls below the threshold, the security level changes [7], as shown in Table 2 the security levels are high, medium, low, and very low. ARSy framework applies security level prediction, based on resource availability.

| Resource | Workload | Security Level | Average Resource (%) |
|----------|----------|----------------|----------------------|
| Maximum  | Light    | High           | 75-100               |
| Maximum  | Light    | Medium         | 50-75                |
| Minimum  | Heavy    | Low            | 20-50                |
| Minimum  | Heavy    | Very Low       | 0-20                 |
3. ARSy Framework Model

To maintain the availability of a resource, researchers have tried to modify the algorithm in use. In particular, [8] 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 [21][14]. The term ARSy Framework is an adaptable resource and security 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 called the Server Data block, which is in a different area from the node, and this research was done until to the delivery of the results of the node as the final destination of all the data.

![ARSy Framework Model Diagram](image)

**Figure 2. ARSy framework [7]**

The client node is a worker node. The process that occurs on the client node is divided into the following process blocks.

a. Data-input block: This block collects data. The collection time was every second in this study. Before the data processed by the data-mining block, the system checks the conditions of the battery, CPU and memory resources in the resource-monitoring block.

b. Resource-monitoring block: This block reports the latest update of the average amount of the sensor node’s resources. This information then is the input for the resource-adaptation block.

c. Resource-adaptation block: This block updates the resource condition with two modes, the first is the status of the resource under adaptation conditions and the second is the status of the resource not under such conditions.

d. Workload-system block: This block provides the workload status of the sensor node if the resources system has a heavy or light workload; heavy workload status means information resources received from resource adaptation blocks are under adaptation conditions, and light workload status means another condition.

e. Resource-, workloads, and security-level-setting block: This block contains the summary of the all the system’s resource conditions, such as the amount of resources that can be used to execute data mining, a security-level status that will be given at data output, and overall system workload.

f. Data-mining block: This block involves mining data based on light-weight frequent item algorithm [22]. The data-mining process carried out by creating counter data. The same data placed on the same counter data and new counter data created for new data types.
g. Data-mining-result block: This block temporarily stores the results of data mining with time-limited, before the Data-mining-result sent to the Security-implementation block.

h. Security-implementation block: This block implements the appropriate security level based on the resource-sensor-node condition, then the results sent to the data server, i.e., the final destination of all data.

4. Implementation and Testing Mechanism

The Raspberry Pi 3 Model B is chosen because it is a single-board model, simple, and lightweight. The model has built-in Wi-Fi, eliminating the need for extra USB Wi-Fi adapters [23]. Another advantage is its compatibility with several operating systems and its plug-and-play compatibility with a variety of equipment.

One challenge of this device involves memory sharing between a CPU and graphics processing unit (GPU) [24]. Some programs are not as demanding on the CPU, and some run on the GPU such as Blu-ray video playback. A GPU is powerful enough to handle applications. Other is with the power-supply-management system. The DS18B20 temperature sensor is a single-wire digital sensor [25] that uses only one cable for communication with the CPU and for grounding. The sensor can derive power directly from the data line.

We conducted our laboratory testing on a single node such as the architecture system shown in Figure 3. The data collected by each node processed with local node resources in accordance with the conditions of the resource node. The output data sent at certain times to the server, which is the final destination of the data.

![Figure 3. Architecture system](image)

The goal of the test was to observe the differences in resource behavior and time efficiency when a security system operates under normal and stressful conditions and implementing an ARSy framework and a non-ARSy framework. We tested under two scenarios: normal and stress. Normal testing conducted by allowing the system to run normally without any intervention or special treatment that would cause the CPU to become busier than usual. Stress testing conducted by making the CPU busier, such as by playing games, browsing websites, and streaming videos. The sensor was touched so that variants data could be collected; otherwise, there would be too few data variants. Stress testing continued until the resources were completely exhausted. The testing parameters are a list in Table 3.

| Table 3. Testing parameters |
|-----------------------------|
| **Parameter**               | **Value**              |
| Critical threshold of Battery, CPU, and Memory | 55% capacity in use |
| Time for data collection    | 1 s                    |
| Time for data release       | 30 s                   |
| Battery capacity            | 30 mAh                 |
| Tests                       | Normal/ Stress         |
| Sensor treatment            | Touched/Untouched      |

5. Results

The resource activities of WSNs are interrelated. In general, battery consumption strongly affected by the activity of the CPU, i.e., busy, normal, or idle. Increased CPU activity increases battery consumption [19]. The discussion of the results of this testing using a parameter in Table 3. The testing divided into two parts, the first by applying the ARSy framework and the second without the
ARSy framework. With ARSy framework carried out with stress tests and normal tests, the same thing also was done without ARSy framework being with stress tests and normal tests.

5.1. **Stress test using ARSy framework**

Results in Table 4 shows the test carried out by conditioning the activity resources to a very busy one so that CPU resource consumption varies. When CPU usage exceeds the threshold limit of 55%, the adaptation system via RF is triggered by reducing the amount of data processed. For example, at time release 120, at that time 80.7% CPU usage had exceeded the threshold limit so that the CPU adaptation policy through Random Factor (RF) would only process 42.89% data. Similar adaption policy is also applied to memory and battery usage. When the memory usage is still smaller than the limit value, the amount of data that will be stored is 50%. If it exceeds the limit, the data stored capacity will be reduced. For example in time release 150, memory usage is 64.67%, then the memory resource adaptation via Radius Threshold (RT) will only save 35.3% of data to memory. Moreover, when battery usage is still smaller than the threshold limit value, the data capture time is done every 1 second. Another, the time capture via Sampling Interval (SI) will increase.

| Time Release (second) | Resource Usage (%) | Resource Adaptation |
|----------------------|-------------------|---------------------|
|                      | CPU   | Memory | Battery | RF (%) | RT (%) | SI (second) |
| 30                   | 57.1  | 38.88  | 4.80    | 95.33  | 50     | 1          |
| 60                   | 49.3  | 54.03  | 1.76    | 0.00   | 50     | 1          |
| 90                   | 71.3  | 57.11  | 31.53   | 63.78  | 42.9   | 1          |
| 120                  | 80.7  | 59.69  | 41.40   | 42.89  | 40.3   | 1          |
| 150                  | 100   | 64.67  | 53.13   | 1.00   | 35.3   | 1.22       |
| 180                  | 78.5  | 68.24  | 68.60   | 47.78  | 31.8   | 1.64       |
| 210                  | 96.4  | 72.84  | 80.07   | 8.00   | 27.2   | 1.96       |
| 240                  | 99    | 76.80  | 91.27   | 2.22   | 23.2   | 2.26       |

Table 5 shows the data security level that processed with the capacity of available resources on the WSN device. Determining the security level of each data output varies based on the average resource, and this affects the status of the workload and adaptation that occurs. This security level policy has been discussing in [7] for high, medium, low and very low-security levels.

| Time Release (second) | Average Resource Usage | Resource Condition | Workload Condition | Status Adaptation | Security Level |
|----------------------|------------------------|--------------------|--------------------|-------------------|----------------|
| 30                   | 33.60                  | MAXIMUM            | LIGHT              | NOT Adaptation    | MEDIUM         |
| 90                   | 53.31                  | MAXIMUM            | LIGHT              | NOT Adaptation    | MEDIUM         |
| 120                  | 60.60                  | MINIMUM            | HEAVY              | Adaptation        | LOW            |
| 180                  | 71.77                  | MINIMUM            | HEAVY              | Adaptation        | LOW            |
| 210                  | 83.09                  | MINIMUM            | HEAVY              | Adaptation        | VERY LOW       |
| 240                  | 89.02                  | MINIMUM            | HEAVY              | Adaptation        | VERY LOW       |

Table 5 shows the data security level that processed with the capacity of available resources on the WSN device. Determining the security level of each data output varies based on the average resource, and this affects the status of the workload and adaptation that occurs. This security level policy has been discussing in [7] for high, medium, low and very low-security levels.

5.2. **Normal Test using ARSy framework**

Table 6 shows different results of normal testing with the ARSy framework. The CPU and memory activity never appear to be busy, because the threshold limit of 55% until the battery run out is never reached. This condition has a good impact on the system because it is more stable and the operating time is longer. The average data output is processed with high and medium security levels as shown in Table 7.
Table 6. Resources activity in a normal test using ARSy framework

| Time Release (second) | Resource Usage (%) | Resource Adaptation |
|----------------------|--------------------|---------------------|
|                      | CPU    | Memory | Battery | RF (%) | RT (%) | SI (second) |
| 30                   | 0.8    | 16.36  | 3.2     | 100    | 50     | 1.00        |
| 90                   | 1.0    | 16.42  | 9.4     | 100    | 50     | 1.00        |
| 150                  | 6.3    | 16.32  | 15.5    | 100    | 50     | 1.00        |
| 270                  | 0.7    | 16.53  | 27.8    | 100    | 50     | 1.00        |
| 330                  | 2.2    | 16.50  | 34      | 100    | 50     | 1.00        |
| 450                  | 2.5    | 16.55  | 46.2    | 100    | 50     | 1.032       |
| 510                  | 1.8    | 16.56  | 52.3    | 100    | 50     | 1.199       |
| 570                  | 1.8    | 16.52  | 58.5    | 100    | 50     | 1.368       |
| 630                  | 1.3    | 16.55  | 64.6    | 100    | 50     | 1.534       |
| 690                  | 1.5    | 16.56  | 70.7    | 100    | 50     | 1.7         |
| 750                  | 1.1    | 16.55  | 76.7    | 100    | 50     | 1.864       |
| 810                  | 0.9    | 16.56  | 82.7    | 100    | 50     | 2.028       |

Table 7. Security level based on resources in a normal test using ARSy framework

| Time Release (second) | Average Resource Usage | Resource Condition | Workload Condition | Status Adaptation | Security Level |
|----------------------|------------------------|--------------------|--------------------|-------------------|---------------|
| 30                   | 6.80                   | MAXIMUM            | LIGHT              | NOT Adaptation    | HIGH          |
| 90                   | 8.93                   | MAXIMUM            | LIGHT              | NOT Adaptation    | HIGH          |
| 150                  | 12.70                  | MAXIMUM            | LIGHT              | NOT Adaptation    | HIGH          |
| 210                  | 13.50                  | MAXIMUM            | LIGHT              | NOT Adaptation    | HIGH          |
| 270                  | 15.00                  | MAXIMUM            | LIGHT              | NOT Adaptation    | HIGH          |
| 330                  | 17.57                  | MAXIMUM            | LIGHT              | NOT Adaptation    | HIGH          |
| 450                  | 21.77                  | MAXIMUM            | LIGHT              | NOT Adaptation    | MEDIUM        |
| 510                  | 23.57                  | MAXIMUM            | LIGHT              | NOT Adaptation    | MEDIUM        |
| 540                  | 24.77                  | MAXIMUM            | LIGHT              | NOT Adaptation    | MEDIUM        |
| 570                  | 25.60                  | MAXIMUM            | LIGHT              | NOT Adaptation    | MEDIUM        |
| 690                  | 29.60                  | MAXIMUM            | LIGHT              | NOT Adaptation    | MEDIUM        |
| 750                  | 31.43                  | MAXIMUM            | LIGHT              | NOT Adaptation    | MEDIUM        |
| 780                  | 32.53                  | MAXIMUM            | LIGHT              | NOT Adaptation    | MEDIUM        |
| 810                  | 33.40                  | MAXIMUM            | LIGHT              | NOT Adaptation    | MEDIUM        |

5.3. Stress test without ARSy framework

The following testing done without ARSy framework and WSNs resources being stress. The results are shown in table 8. Operating time is only 210 seconds, shorter than table 4, which is up to 240 seconds.

Table 8. Security level based on resources in stress test without ARSy framework

| Time Release (second) | Resources Usage (%) | Resources Adaptation |
|----------------------|--------------------|----------------------|
|                      | CPU    | Memory | Battery | RF (%) | RT (%) | SI (second) |
| 30                   | 59.4   | 29.92  | 9.63    | -      | -      | -          |
| 90                   | 89.7   | 56.53  | 37.37   | -      | -      | -          |
| 120                  | 83.2   | 59.24  | 52.83   | -      | -      | -          |
| 150                  | 78.5   | 61.49  | 68.30   | -      | -      | -          |
| 180                  | 71.6   | 62.13  | 77.90   | -      | -      | -          |
| 210                  | 76.5   | 62.55  | 89.37   | -      | -      | -          |

Because this test designed without an adaptation mechanism, the results in table 8 are obvious when the resource exceeds the threshold limit of 55%, so there is no resource adaptation mechanism.
Moreover, the results are shown in Table 9 where all data processed with High-level security. With these conditions, the impact on operational time is shorter without ARSy framework, when compared to the testing using the ARSy framework.

**Table 9. Resources activity in stress test without ARSy framework**

| Time Release (second) | Average Resource Usage | Resource Condition | Workload Condition | Resources Status | Security Level |
|-----------------------|------------------------|--------------------|--------------------|------------------|----------------|
| 30                    | 32.97                  | MAXIMUM            | LIGHT              | NOT Adaptation   | HIGH           |
| 90                    | 61.18                  | MINIMUM            | HEAVY              | NOT Adaptation   | HIGH           |
| 120                   | 65.07                  | MINIMUM            | HEAVY              | NOT Adaptation   | HIGH           |
| 150                   | 69.43                  | MINIMUM            | HEAVY              | NOT Adaptation   | HIGH           |
| 180                   | 70.53                  | MINIMUM            | HEAVY              | NOT Adaptation   | HIGH           |
| 210                   | 76.12                  | MINIMUM            | HEAVY              | NOT Adaptation   | HIGH           |

5.4. Normal test without ARSy framework

The following testing is still without ARSy framework and allows the resource to work normally. The results are very significant in terms of the time operation as shown in Table 10, which operates only up to 330 seconds. Moreover, the average resource used is very small as shown in Table 11.

**Table 10. Security level based on resources in the normal test without ARSy framework**

| Time Release (second) | Resources Usage (%) | Resources Adaptation |
|-----------------------|---------------------|----------------------|
|                       | CPU                 | Memory               | Battery            | RF (%) | RT (%) | SI (second) |
| 30                    | 1                   | 16.16                | 8.03               | -      | -      | -           |
| 90                    | 1                   | 16.22                | 24.30              | -      | -      | -           |
| 150                   | 0.8                 | 16.23                | 40.83              | -      | -      | -           |
| 210                   | 1.3                 | 16.23                | 57.10              | -      | -      | -           |
| 270                   | 1.3                 | 16.24                | 73.63              | -      | -      | -           |
| 330                   | 0.8                 | 16.24                | 89.90              | -      | -      | -           |

**Table 11. Resources activity in the normal test without ARSy framework**

| Time Release (second) | Average Resource Usage | Resource Condition | Workload Condition | Resources Status | Security Level |
|-----------------------|------------------------|--------------------|--------------------|------------------|----------------|
| 30                    | 8.41                   | MAXIMUM            | LIGHT              | NOT Adaptation   | HIGH           |
| 90                    | 13.83                  | MAXIMUM            | LIGHT              | NOT Adaptation   | HIGH           |
| 150                   | 19.28                  | MAXIMUM            | LIGHT              | NOT Adaptation   | HIGH           |
| 210                   | 24.87                  | MAXIMUM            | LIGHT              | NOT Adaptation   | HIGH           |
| 270                   | 30.38                  | MAXIMUM            | LIGHT              | NOT Adaptation   | HIGH           |
| 330                   | 35.63                  | MAXIMUM            | LIGHT              | NOT Adaptation   | HIGH           |

6. Conclusion and Future Work

The most interesting thing in this study is to conserve the WSNs resource availability as it will be implemented in extreme areas where there is no possibility of maintenance on the device. Therefore, the available resources must be used as efficiently as possible until the available resources are completely used up. The tests carried out are still limited to the lab test, concluded as follows: Firstly, the resource is more efficient in data processing by applying mining data to extract the most dominant data types that will be sent to the server. The time operation of resource adaptation via ARSys framework is longer than without resource adaptation. Resource and security adaptation is able to maintain resource availability and data quality before sending to the server. Finally, to maintain the sustainability of the system, the availability of energy batteries at WSNs is the main goal.
References

[1] T. Rault, A. Bouabdallah, and Y. Challal, “Energy efficiency in wireless sensor networks: A top-down survey,” Comput. Networks, vol. 67, pp. 104–122, 2014.

[2] A. A. Habib F. Rashvand., “Wireless Sensor Systems for Extreme Environments,” in John Wiley & Sons Ltd, John Wiley & Sons Ltd, 2017.

[3] I. F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci, “A survey on sensor networks,” IEEE Commun. Mag., vol. 40, no. 8, pp. 102–105, 2002.

[4] J. M. Parenreng and A. Kitagawa, “Resource Optimization Techniques and Security Levels for Wireless Sensor Networks Based on the” Sensors (Basel), 2018.

[5] R. Uwe and Q. Tse, “Enabling Resource-Aware for In-network Data Processing in Wireless Sensor Networks,” in Conferences in Research and Practice in Information Technology (CR-PIT), 2008.

[6] A. M. Shiddiqi, “RESOURCE-AWARE DATA STREAM CLASSIFICATION IN WIRELESS SENSOR NETWORKS,” no. July, 2009.

[7] J. M. Parenreng and A. Kitagawa, “A Model of Security Adaptation for Limited Resources in Wireless Sensor Network,” J. Comput. Commun., vol. 5, no. 3, pp. 10–23, 2017.

[8] M. M. Gaber and P. S. Yu, “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, pp. 649–656, 2006.

[9] N. K. Pour, “Energy Efficiency in Wireless Sensor Networks,” no. December, 2016.

[10] B. Ksiezopolski, P. Szalachowski, and Z. Kotulski, “SPOT: Optimization Tool for Network Adaptable Security,” Commun. Comput. Inf. Sci., pp. 269–279, 2010.

[11] B. Ksiezopolski and Z. Kotulski, “Adaptable security mechanism for dynamic environments,” Comput. Secur., vol. 26, no. 3, pp. 246–255, 2007.

[12] I. F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci, “Wireless sensor networks: a survey,” Comput. Networks, vol. 38, no. 4, pp. 393–422, 2002.

[13] P. Archana, Bharathidasan., Vijay, Anand, Sai, “Sensor networks: an overview,” IEEE Potentials, vol. 22, no. 2, pp. 20–23, 2003.

[14] J. Parenreng, M. I. Syarif, S. Djanali, and A. M. Shiddiqi, “Performance analysis of resource-aware framework classification, clustering and frequent items in wireless sensor networks,” Proceeding Int. Conf. eEducation Entertain. eManagement, pp. 117–120, 2011.

[15] N. D. Phang, M. M. Gaber, and U. Röhm, “Resource-aware Online Data Mining in Wireless Sensor Networks,” Pro- ceedings IEEE Symp. Comput. Intell. DataMining, no. Cidm, pp. 139–146, 2007.

[16] L. Caviglione and A. Merlo, “The energy impact of security mechanisms in modern mobile devices,” Netw. Secur., vol. 2012, no. 2, pp. 11–14, 2012.

[17] B. Ksiezopolski and Z. Kotulski, “Adaptive Approach to Network Security,” CCIS, pp. 233–241, 2009.

[18] V. Raghunathan, C. Schurgers, S. Park, and M. B. Srivastava, “Energy-aware wireless microsensor networks,” IEEE Signal Process. Mag., vol. 19, no. 2, pp. 40–50, 2002.

[19] N. R. Potlapally, S. Ravi, A. Raghunathan, and N. K. Jha, “A study of the energy consumption characteristics of cryptographic algorithms and security protocols,” IEEE Trans. Mob. Comput., vol. 5, no. 2, pp. 128–143, 2006.

[20] J. Sen, “A Survey on Wireless Sensor Network Security,” Comput. Networks, vol. 1, no. 2, pp. 55–78, 2009.

[21] A. M. Shiddiqi, “Performance Measurement of Resource-aware Framework in Online Data Stream Mining,” pp. 1–7, 2007.

[22] J. Parenreng, S. Djanali, and A. M. Shiddiqi, “ANALISA KINERJA RESOURCE-AWARE FRAMEWORK PADA ALGORITMA LIGHT-WEIGHT FREQUENT ITEM ( LWF ),” Proceeding SNPI ITS Surabaya, 2010.

[23] S. Monk, Raspberry Pi Cookbook, no. December. 2015.

[24] S. McManus, “Raspberry Pi For Dummies,” in John Wiley & Sons, Inc. 2014. p. 456.

[25] Maxim Integrated, “Datasheet DS18B20,” Maxim Integr., vol. 92, no. 20, 2015.