






Evaluation of a Robust Fault-Tolerant Mechanism for Resilient IoT Infrastructures

José Manuel Lozano Domínguez[✉] ,
Tomás de J. Mateo Sanguino , and Manuel J. Redondo González 

University of Huelva, Dpto. Ing. Electrónica, de Sistemas Informáticos
y Automática, Ctra Huelva - La Rábida S/N, 21819 Palos de la Frontera, Spain
josemanuel.l Dominguez@alu.uhu.es

Abstract. Gateways in IoT infrastructures generally represent a single point of failure, thus resulting in a total loss of network operability. This paper presents the design, implementation and experimentation of a fault-tolerant protocol for a critical infrastructure applied to the field of road safety. The proposed mechanism establishes a node hierarchy to prevent loss of communication against AP failures in WLANs based on the IEEE 802.11n standard. This mechanism automates the management of the node roles by means of an election and promotion process between stations in search of designated and backup APs. The convergence times of the protocol obtained suitable values of 3.34 s for the formation of a BSS from zero, as well as 15.20 s and 18.84 s for the failover conditions of the backup and designated APs with a minimum traffic load of 42.76% over the WSN traffic.

Keywords: Failover mechanism · Fault tolerance · IoT · Resilience
WSN

1 Introduction

Typically, IoT infrastructures communicate through a central node or gateway that serves as a connection point between sensors, controllers and the outside [1]. However, this represents a single point of failure that diminishes the availability and reliability required by critical applications such as health monitoring [2], cybersecurity in infrastructures [3, 4] or personal safety [5]. The state of the art on protocols related to IoT and WSN applications have been mainly designed to improve the performance and hierarchy of networks. For instance, the improvements studied in [6] focus on the automation of the management and maintenance of tasks, as well as on increasing robustness under failures (e.g., electrical or communication). Thus, the use of third-party management protocols (e.g., SNMP) was proposed in [7] to monitor the node status and send warning messages. Moreover, a local self-recovery mechanism based on flash memories was proposed in [8] to prevent data transfer and network load. In [9, 10] several ways to avoid the loss of communication between a cluster and the outside were addressed, where the gateway is selected according to battery levels. Also for this purpose, in [11] is described a solution to detect failures (e.g., low energy thresholds) and manage gateways locally to avoid loss of communication of a WSN using virtual cells or groups of

nodes. Similarly, a cluster-head structure consisting of cell-head nodes is organized to communicate with a base station depending on the sensors' energy [12]. Moreover, the works described in [13, 14] focus on restoring the communication and retrieving information between a node and its gateway designating new routes through backup clusters. Also, [15] describes a technique that offers fault tolerance in large IEEE 802.11 infrastructures working in an ESS topology. This technique uses an algorithm that structures the network using the coverage and performance criteria in such a way that, when an AP fails, a new route is searched to reach any point of the network through a Spanning Tree Protocol [16].

The protocol presented in this paper has been implemented as part of an intelligent object detection and signaling system applied to road safety [17]. The goal of the system, consisting of a set of autonomous sensing devices, is to interact with the environment to distinguish vehicles, generate visual alerts in the presence of pedestrians on zebra crossings and help reducing road accidents. Each device comprises a unit based on a microcontroller and a wireless communication module responsible for sending and receiving pedestrian detection messages to activate the light signaling units. To this end, a WSN infrastructure of nodes that work in a coordinated way within a BSS has been implemented. One of the sensing devices has the role of AP, whose function is to manage and control the network operation. The rest of the devices are clients in such a way that when one of them detects a pedestrian, it sends a broadcast message to the rest of the devices through the WLAN. Said communication system has the function, therefore, of synchronizing a light signaling barrier over the road. Due to its critical mission, the proposed mechanism has the objective of preventing the loss of communication in the WLAN through a redundancy and high availability strategy based on a hierarchy of nodes that act as APs.

This paper is structured as follows: Sect. 2 describes the protocol as well as its operation; Sect. 3 shows the experimentation carried out and the results. Finally, Sect. 4 presents the conclusions and future work.

2 Protocol Description

The protocol presented in this paper is applied to IoT infrastructures based on WLAN nodes operating through a BSS. More specifically, the communication is based on the IEEE 802.11n standard [18] working at 300 Mbps, which uses WPA/WPA2 encryption [19], IPv4 unicast/broadcast packets at the network layer, and UDP datagrams at the transport layer since it accelerates the message delivery with regard to TCP by dispensing with ACK messages as discussed in [20].

The protocol has been designed in the application layer of the OSI model and its objective is to prevent an IoT infrastructure from running out of communication due to faults in the central AP (e.g., power failures). This is possible thanks to a high availability structure consisting of a designated AP (APd), a backup AP (APb) and client nodes. The APd is responsible for coordinating the delivery of data frames at the MAC level, the APb aims to assume the functions of the APd in case of failure and the clients have the capability to auto reconfigure themselves as APd or APb.

The protocol autonomously manages the node roles and responds to changes in the WLAN structure through an active exchange of messages. In brief, the operation is as follows: (1) initially, the node that acts as AP is designated; (2) secondly, the node that acts as APd or APb is determined based on the existing neighbors; (3) the adjacency between APd and APb is maintained by bidirectional hello messages sent periodically; (4) in case of losing adjacency after n hello messages, a promotion process is initiated between the nodes to assign new roles according to the case (i.e., APd, APb or both). To do this, the protocol also manages the automatic IP addressing via a DHCP server. Thus, the station set as APd takes the first IP address of the network (e.g., 192.168.0.1) and the rest of nodes receive consecutive IP addresses from the available pool. When the APd node changes, all clients have to reconnect to the new APd node and renew their IP addresses.

The protocol establishes a set of phases through which each node must jump until the roles converge. These are divided into a preliminary phase that determines if a node should act in AP or station mode (Fig. 1) along with three subsequent phases called init, stability and maintenance (Fig. 2). Thus, each time a node joins the WLAN, it first checks if there is an AP to which to link and request the network status. In negative case, the node is set to AP mode, waits for new incoming clients and then goes to the init phase.

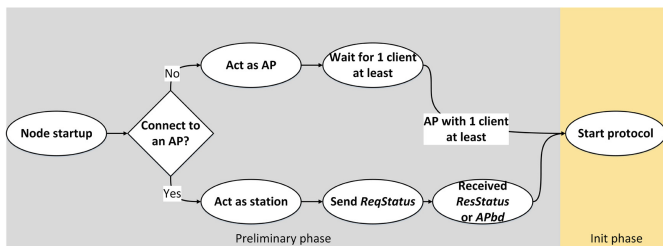


Fig. 1. Description of a node startup process and protocol launching.

2.1 Init Phase

This phase aims to determine which WLAN nodes act as APd and APb as they join the BSS. Each node initiates a competition consisting in exchanging messages with the neighbors considering a priority value—provided under the network administrator’s criteria—and the MAC address fields as follows: (1) the node with lowest priority will have the highest probability of being APd; (2) the node with the next lowest priority will be candidate for APb; (3) in case of tie, the minor MAC address is taken as criteria; (4) when the information converges, the rest of the WLAN is informed by identifying the APb and APd nodes. In detail, the different tasks that each node can carry out in this phase are the following.

First Message. When a node initiates the protocol and has no information received from the network yet, it sends a first message including its own MAC address, the priority values to be candidate for APd and APb, and the default interval to exchange the adjacency messages, among other fields.

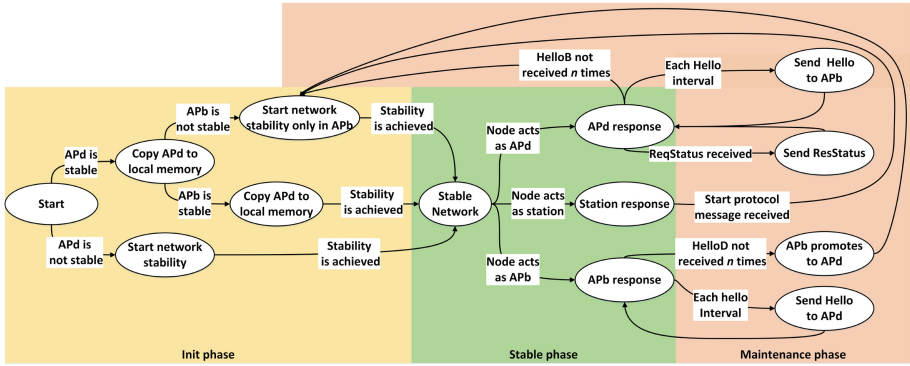


Fig. 2. Operating states of the protocol.

Message Received. When a node receives a message, it processes the data and compares the information received with the information locally stored.

Selection Mechanism. For each message received, a node inspects the priority values and MAC addresses to learn or replace their values in the APd or APb fields if better.

Message Sent. Each node forwards update messages to the WLAN identifying the nodes that are being promoted to APd and APb.

Hello Update. With each periodical message received identifying the APd of the WLAN, the nodes locally update the adjacency intervals to ensure the concordance between APd and APb.

Wait Time. The init phase is repeated in a loop until reaching the convergence or until expiring a timer. At the end of this loop, a node passes to the stability phase (Fig. 3).

2.2 Stability and Maintenance Phases

The stability phase remains idle all the time when the roles of the APd, APb or the clients are assigned, but the nodes leave it to enter the maintenance phase if a network event occurs. The possible events consist in receiving messages to request the network status, exchange protocol information or maintain adjacencies, as well as managing the expiration of the waiting times. In the latter case, the nodes return to the init phase to search for new APb or APd nodes according to the case.

2.3 Message Structure

The protocol establishes four different messages: (a) status request, (b) status response, (c) exchange of protocol information, and (d) adjacency.

The status request message (*ReqStatus*) consists of 9 bytes, where the first 8 bytes include the identification tag and the last one indicates the protocol version (Fig. 4a). This message is sent by a client node connected to the AP that remains in station mode during the init phase. The status response message (*ResStatus*) and the exchange of

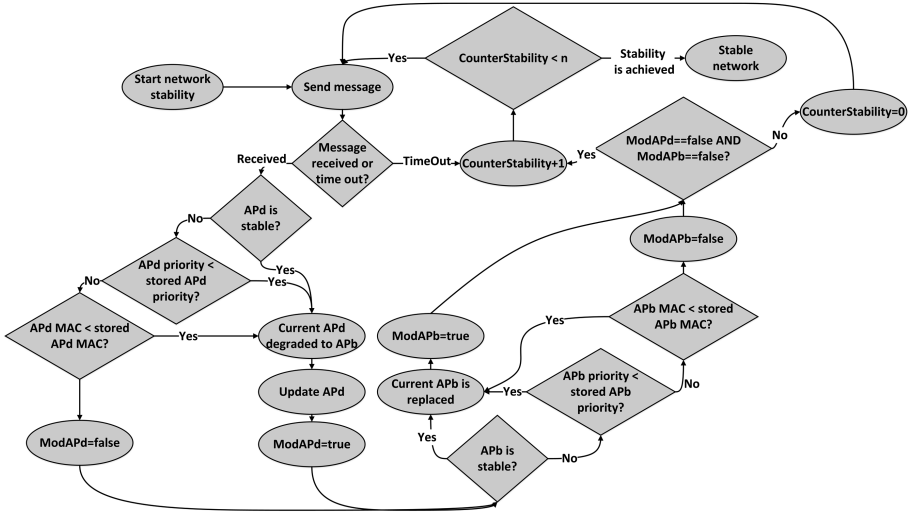


Fig. 3. Diagram of the init phase designed to determine the APd and APb nodes.

protocol information messages (*APdb*) consist of 23 bytes and 19 bytes respectively, matching the same fields except for the first field to differentiate the type of request (Fig. 4b). While the *ResStatus* message can only be sent from the APd node to the unicast address of the client that requested the information on the network state, the *APdb* message can be sent from all the network nodes to a broadcast address to determine which node will act as APd and APb. The rest of the fields contain the following labels: *APdID* and *APbID* include the MAC address of the APd and APb nodes, *APd Priority* and *APb Priority* establish the precedence of the APd and APb nodes, *WLAN Area* identifies the community to which the message belong, *Hello Interval* codifies the time gap in seconds (Table 1), *Network Status* indicates the state of the network (Table 1) and *Version* stands for the form of protocol used.

8 bytes	1 byte	6 bytes	1 byte	6 bytes	1 byte
<i>ReqStatus</i>	Version	<i>HelloD</i>	Version	<i>HelloB</i>	Version
(a)		(c)		(d)	

4 / 8 bytes	6 bytes	6 bytes	1 byte		1 byte		1 byte	
32 / 64 bits	48 bits	48 bits	4 bits	4 bits	4 bits	2 bits	2 bits	8 bits
<i>APdb / ResStatus</i>	APd ID	APb ID	APd priority	APb priority	WLAN area	Hello interval	Network status	Version
(b)								

Fig. 4. Protocol message formats: (a) status request (*ReqStatus*), (b) exchange of protocol information (*APdb*) or status response (*ResStatus*), (c) and (d) adjacency maintenance (*Hello*).

Table 1. Hello interval and network status coding.

Binary coding	Decimal coding	Hello intervals by default	Network status
00	0	3, 5, 10 & 20 s	Network no stable
01	1	3, 5, 10 & 20 s	APd stable & APb no stable
10	2	3, 5, 10 & 20 s	APd stable & APb stable
11	3	3, 5, 10 & 20 s	Reserved

The adjacency message consists of 7 bytes, where the first 6 bytes include the *HelloD* or *HelloB* tags to identify the source node and the last byte indicates the protocol version (Fig. 4c and d).

2.4 Message Exchange

The APd node alternates the stability and maintenance phases after the following events: (i) a *ReqStatus* message arrives, which is answered with a *ResStatus* message; (ii) the *HelloD* interval expires, initiating the delivery of a new adjacency message; (iii) a *HelloB* message arrives from the APb, indicating adjacency; (iv) the timer expires after n consecutive intervals with no *HelloB* messages, thus indicating loss of adjacency with the APb node. In this case, a procedure to send an *APdb* message is initiated to start the search for a new node that acts as backup AP.

Moreover, the APb node leaves the idle state and enters the maintenance phase after the following cases: (i) a *HelloD* message arrives from the APd, indicating adjacency; (ii) the *HelloB* interval expires, initiating the delivery of a new adjacency message; (iii) the timer expires after n consecutive intervals with no *HelloD* messages, thus indicating loss of adjacency with the APd node. In this case, the APb node promotes itself to APd and initiates a procedure by sending an *APdb* message to start the search for a new node that acts as backup AP.

Finally, the nodes acting as stations remain in the stability phase until receiving an *APdb* message. Then, a contention process between the WLAN nodes is initiated to designate a new backup AP.

3 Experimentation

The experimentation carried out consisted in measuring the convergence time and traffic load of the WSN in three representative case studies: (i) establishing the BSS from zero, (ii) after APb failover, and (iii) after APd failover. To this end, a WSN with 6 nodes was deployed in a real working scenario. The network stability was perturbed to trigger the contention mechanism and promote the nodes to APd, APb or remain as stations. We used the ESP8266 microcontroller from Espressif Systems Ltd. The code size required to store the developed protocol occupied 261 KB, standing for the 26.1% of the flash memory.

The protocol was configured to establish the network convergence after receiving $m = 3$ APdb messages and to lose the adjacency after mislaying $n = 5$ consecutive hello messages (i.e., *HelloD* or *HelloB*). An analysis performed with Acrylic[®] WiFi Home showed 41 neighboring APs coexisting in the working environment, of which 24 APs were in adjacent Wi-Fi channels with a separation less than 30 MHz. The RSSI of the AP deployed was -49.05 ± 6.28 dB, while the average RSSI of the inferring channels was -63.18 ± 7.20 dB. As a result, the air quality of the BSS obtained an equivalent score of 3 out of 10.

Establishing the BSS from Zero In this case, there was no APd or APb operating previously in the network. The condition to achieve the stability by the AP and the rest of the nodes was measured considering a series of 10 samples, counting the total time (t_z) from the first *ReqStatus* message received in the network until the end of the *APdb* message exchange (Fig. 5a). The standard deviation observed in Table 2 indicates the co-channel interference level due to other APs in the area. This caused collisions between the frames and therefore the forwarding of new messages.

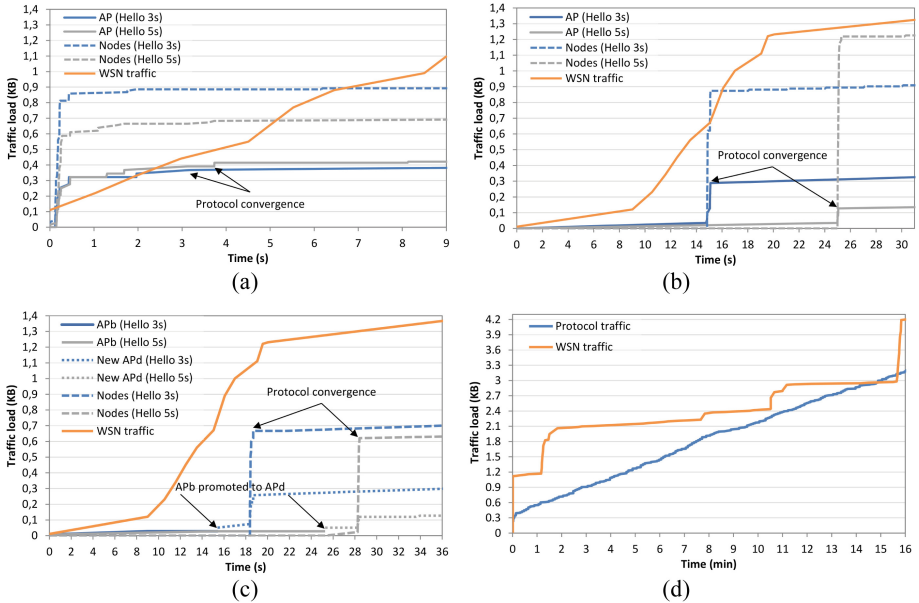


Fig. 5. Accumulative traffic load in short and long terms: (a) establishing the BSS from zero, (b) APb failover, (c) APd failover, and (d) protocol traffic vs background traffic.

APb Failover. In this scenario, the BSS was already operative when the APb was induced to fail, so it stopped issuing *HelloB* messages. Subsequently, the APd sent an *APdb* message to activate the mechanism once the loss of adjacency with the APb was detected and the rest of the nodes competed for being the following APb. The convergence time (t_b) was measured from the last hello message successfully received from the APb until the end of the *APdb* message exchange (Fig. 5b). The time resulted in the expression $t_b = n \times t$, where the hello interval was set to $t = 3$ s (Table 3).

Table 2. Convergence values establishing the BSS from zero.

Test	Time (s) for t_z	Frames	AP traffic (%)	Node traffic (%)
1	3.19	60	26.67	73.33
2	4.81	53	28.30	71.70
3	3.19	49	26.53	73.47
4	3.28	45	33.33	66.67
5	3.11	65	24.62	75.38
6	3.12	58	28.81	71.19
7	3.14	59	28.81	71.19
8	3.22	81	19.75	80.25
9	3.18	63	26.98	73.02
10	3.15	82	14.63	85.37
Average	3.34 ± 0.52	61.50 ± 12.19	24.72 ± 5.13	75.28 ± 5.13

Table 3. Convergence values after APb & APd failovers (hello interval every $t = 3$ s).

Test	Time (s) for t_b t_d	Frames	AP traffic (%)	Node traffic (%)
1	15.09 18.68	54 49	29.63 22.45	70.37 77.55
2	15.06 18.54	48 66	37.50 21.21	62.50 78.79
3	15.06 18.72	46 106	34.78 7.55	65.22 92.45
4	15.27 18.55	70 68	28.57 20.59	71.43 79.41
5	15.26 18.56	50 86	30.00 18.60	70.00 81.40
6	15.48 21.52	82 76	12.20 25.00	87.80 75.00
7	15.09 18.47	60 53	23.33 9.43	76.67 90.57
8	15.16 18.26	51 20	31.37 30.00	68.63 70.00
9	15.27 18.68	70 85	28.57 15.29	71.43 84.71
10	15.26 18.42	74 83	29.73 10.84	70.27 89.16
Average	15.20 ± 0.13 18.84 ± 0.95	60.5 ± 12.62 69.2 ± 24.10	27.60 ± 6.88 16.62 ± 7.24	72.40 ± 6.88 83.38 ± 7.24

APd Failover. In this case, the BSS was already operative when a failure was induced in the APd, so it stopped providing the WLAN service and did not send *HelloD* messages. When the APb detected that the adjacency with the APd ended after the loss of *HelloD* messages, it promoted itself as APd and sent an activation *APdb* message so that the rest of nodes would compete for being the following APb. The convergence time (t_d) was measured from the last hello message successfully received from the APd until the end of the *APdb* message exchange (Fig. 5c). The convergence time resulted in the combination of $t_d = t_z + t_b$ as shown in Table 3.

Figures 5a-5d show the cumulative traffic load of the protocol with respect to time considering a set of hello messages sent every $t = 3$ s and 5 s. The typical traffic generated by the WSN under normal operating conditions, as described in [17], has been included with the goal of evaluating the impact of the *APdb* protocol on the network. The behavior was measured in short and long term. From the traffic response

(Fig. 5a–c), it is observed that the hello interval (t) does not modify the protocol pattern except that the sequence of messages exchanged between the nodes is simply shifted in time. As shown in Fig. 5d, the WSN traffic vs APdb protocol traffic required 4311 bytes and 3220 bytes respectively after 16 min of operation. This represents a 42.76% of traffic load with respect to the total traffic generated, thus presenting an assumable impact on the WSN operation.

4 Conclusions

Typically, IoT infrastructures communicate through a central node that serves as a connection point between sensors, actuators and the outside. When said node fails, it compromises reliability, endangering the entire network. To avoid it, protocols must automate the network management and provide high tolerance to failures (e.g., electrical, communication, etc.).

With this aim, this paper proposes a protocol that autonomously manages a high availability node-based structure for critical WSN applications based on microcontroller. For this purpose, a promotion and adjacency maintenance process between neighbors has been designed using bidirectional control and hello messages whose goal is to designate a main AP (APd) and a backup AP (APb) from the WSN. This voting process combines MAC addresses and priority values configurable for each node to gain flexibility. The experimentation carried out has shown that the protocol is robust in a real scenario with co-channel interferences (i.e., 3/10 air quality) and consistent to changes in the hello messages and intervals (n , m , t). With the proposed default values, the tests showed that the protocol generates a lower traffic load (42.76%) than the WSN background traffic with acceptable convergence times to establish a BSS from zero ($t_z = 3.34 \pm 0.52$ s), to detect and re-establish the backup AP ($t_b = 15.20 \pm 0.13$ s), as well as to detect and restore the main AP ($t_d = t_z + t_b = 18.84 \pm 0.95$ s).

Future works will be focused on optimizing the delivery of adjacency messages between the designated and backup APs according to battery power levels. In this way, the higher the available energy the larger will be the intervals between messages (i.e., less likely to suffer a power failure) and vice versa, being able to increase the WSN life cycle based on the energy as criteria. Other future works are oriented to validate the protocol scalability in large and populated WSNs (i.e., more than 50 nodes) by means of simulations based on the analysis with intelligent agents.

References

1. Zanella, A., Bui, N., Castellani, A., Vangelista, L., Zorzi, M.: Internet of things for smart cities. *IEEE Internet Things J.* **1**(1), 22–32 (2014)
2. Ortiz, K.J.P., et al.: IoT: Electrocardiogram (ECG) monitoring system. *Indonesian J. Electr. Eng. Comput. Sci.* **10**(2), 480–489 (2018)
3. Alonso, L., Barbarán, J., Chen, J., Díaz, M., Llopis, L., Rubio, B.: Middleware and communication technologies for structural health monitoring of critical infrastructures: a survey. *Comput. Standards Interfaces* **56**, 83–100 (2018)

4. Samboni, F., et al.: MEC IoT: monitorización de estructuras civiles en el contexto IoT. In: Colombian Conference on Communications and Computing, pp. 1–6, Cartagena (2017)
5. Helen, A., Fathila, F., Rijwana, R., Kalaiselvi, V.K.G. A smart watch for women security based on IoT concept ‘watch me’. In: International Conference on Computing and Communications Technologies, pp. 190–194, Chennai (2017)
6. Ray, P.P., Mukherjee, M., Shu, L.: Internet of things for disaster management: state-of-art and prospects. *IEEE Access* **5**, 18818–18835 (2017)
7. Gautam, B.P., Wasaki, K., Sharma, N. A Novel Approach of Fault Management and Restoration of Network Services in IoT Cluster to Ensure Disaster Readiness. In: Networking and Network Applications, pp. 423–428, Hakodate (2016)
8. Pan, W.M.: Dynamic update mechanism in wireless sensor Networks. *Appl. Mech. Mater.* **526**, 267–272 (2014)
9. Din, S., et al.: Energy efficient topology management scheme based on clustering technique for software define wireless sensor network. In: Peer-to-Peer. Springer US (2017). <https://doi.org/10.1007/s12083-017-0607-z>
10. Xian, T.: A modified energy efficient backup hierarchical clustering algorithm for WSN. In: Information Security and Control, pp. 45–48, Taiwan (2012)
11. Asim, M., Mokhtar, H., Merabti, M.: A Cellular approach to fault detection and recovery in wireless sensor networks. In: Sensor Technologies and Applications, pp. 352–357, Glyfada, Athens (2009)
12. Yektaparast, A., Nabavi, F.H., Sarmast, A.: An improvement on LEACH protocol (Cell-LEACH). In: International Conference on Advanced Communication Technology, pp. 992–996, Pyeong Chang (2012)
13. Goratti, L., Kato, S.N.A.C.R.P., et al.: A connectivity protocol for start topology wireless sensor network. *IEEE Wirel. Commun. Lett.* **5**, 12–123 (2016)
14. Induja, K., Deva Kupra, A.J.: A connectivity protocol for star topology using wireless sensor network. In: Nextgen Electronic Technologies: Silicon to Software, pp. 50–56, Chennai (2017)
15. Bhoi, S.K., Panda, S.K., Khilar, P.M.: A network survivability approach to resist access point failure in IEEE 802.11 WLAN. In: Sathiakumar, S., Awasthi, L., Masillamani, M., Sridhar, S. (eds.) Proceedings of International Conference on Internet Computing and Information Communications. Advances in Intelligent Systems and Computing, vol. 216. Springer, New Delhi (2014). https://doi.org/10.1007/978-81-322-1299-7_28
16. Sfeir, E., Pasquahi, S., Schwabe, T., Iselt, A.: Performance evaluation of ethernet resilience mechanisms. In: Workshop on High Performance Switching and Routing, pp. 356–360, Hong Kong, China (2005)
17. Lozano Domínguez, J.M., Mateo Sanguino, T.J.: Design, modelling and implementation of a fuzzy controller for an intelligent road signaling system. *Complexity*, 2018, Article ID 1849527 (2018)
18. IEEE Computer Society: IEEE Standard for Information Technology–Telecommunications and Information Exchange Between Systems Local and Metropolitan Area Networks–Specific requirements - Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications. Technical Report (2016)
19. Adnan, A.H., et al.: A comparative study of WLAN security protocols: WPA, WPA2. In: Advances in Electrical Engineering, pp. 165–169, Dhaka (2015)
20. Masirap, M., et al: Evaluation of reliable UDP-based transport protocols for internet of things (IoT). In: IEEE Symposium Computer Applications & Industrial Electronics, pp. 200–205, Penang (2016)