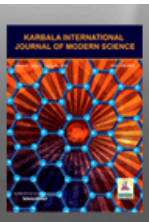

Karbala International Journal of Modern Science

Volume 7 | Issue 4 Article 14

Analysis of Modern Communication Protocols for IoT applications

Neerendra Kumar
Department of Computer Science and IT, Central University of Jammu, India, neerendra.csit@cujammu.ac.in

Pragti Jamwal
Department of Computer Science and IT, Central University of Jammu, India

Analysis of Modern Communication Protocols for IoT applications

Abstract

To facilitate increasing interactions of machines, various protocols are building up for IoT applications. Safeguarding communication among devices is necessary. In this paper, various protocols have been studied and compared using six parameters: Communication overhead, Security, Packet Loss, Throughput, Bandwidth and Support to QoS. LIDOR is found as the most successful communication protocol. A tabulated comparison of various communication protocols have been provided by comparing the parameters. Further, six star rating for the various protocols have been proposed. The study shows that LIDOR increases reliability under DoS attacks. Considering the existing literature, new research gaps and challenges have been presented.

This research paper is available in Karbala International Journal of Modern Science: https://kijoms.uokerbala.edu.iq/home/vol7/iss4/14

1. Introduction

The Internet of Things is a collection of technologies and standards. IoT allows Internet links to be extended to every type of smart object within the domain. IoT can be described as anything that has a sensor attached to it and can transmit data over the internet from one computer to another or even to a person. In simple words, IoT means that remote computers, machines, people, the environment communicate closely in real-time.

However, in the last ten years, the term IoT has included a wider range of applications such as healthcare, utilities, and transportation, etc. [2]. The applications of IoT technologies are many because IoT applies to any or all techniques that are capable of providing relevant data concerning the operation and performance of the associated activity. In addition, IoT relates to the environmental conditions that are required to observe and can be managed at a distance. The IoT-connected smart devices are more vulnerable to attacks. It's also critical to adopt the correct protocols to minimize security flaws. IoT communication protocols are forms of communication that ensure the data being transmitted between IoT-linked devices. An IP (Internet Protocol) network or a non-IP network can be used to connect IoT devices. Even though the range, power, memory usage of these devices are different. Wireless sensors, software, computing devices, and actuators are different types of IoT products. These are connected to a particular entity on the Internet. It permits data to move automatically among people or objects without the necessity of human interaction. Following the study given in Ref. [1], the IoT architecture has been presented in Fig. 1.

Sensors gather data, which is then processed by device software before being sent to actuators for action. Numerous IoT devices are low-cost and have limited resources, like network connectivity, electricity, and processing capability [3]. To arrange data streams and IoT connections with one another, communication protocols are crucial. IoT security is becoming more of an issue. The Internet of Things is expected to bind trillions of devices around the world in the future. When it comes to secrecy, sender and receiver device's authenticity, and integrity of conveyed messages, the transmission of data among the devices must be secure [4]. Estimates indicate that the count of global IoT devices has surpassed the global population reaching 8.4 billion, back in 2017 only. The number of IoT devices on the planet is expected to exceed 20.4 billion by 2020. According to Huawei, by 2025, the count of IoT devices will be approximately near 100 billion, with 2 million sensors installed per hour [5].

In the tumultuous world of IoT, privacy and security will be a major concern. Malicious users can access the attack surfaces which include communication channels between IoT devices, also between devices and the back-end infrastructure, back-end data storage, and IoT-specific back-end applications [6].

Agriculture, e-health, smart grid, Intelligent transportation networks, smart cities, and hospitality are only a few of the exciting applications for IoT. The sources of data are numerous. Sources are increasing day by day producing big data. Big data is characterized not only by the data but also by complexity [7]**.** IoT-enabled systems have become a reality due to the growing presence of technology in our daily lives due to the increasing importance of connectivity in human life. Physically harming a remote computer or 'sniffing and altering sensitive data' may be disastrous. The protection of such reasonable data from malicious attacks is critical. Therefore, the protection of data becomes a current topic of research in the IoT area. In terms of safety, Internet of Things (IoT) applications have become inextricably connected to our daily lives. These systems need wireless networking solutions which are: (i) secure and robust (ii) scalable and (iii) supply low latency. The use of low-overhead routing strategies for multi-hop communication is the secret to scalability [8]**.** Following [9], a detailed IoT analytics has been provided in Fig. 2.

By defending against attacks on availability, security and robustness can be achieved. In an IoT network, an IoT device's intimate delivery of information over the network may be a matter of the utmost importance. The need of safeguarding IoT connections cannot be emphasized. Safeguarding is especially required with expanding the number of IoT devices and instances, as well as the increasing reliance on their use. Recently, concerns about a wide range of vulnerabilities in IoT communications have been growing. The Internet of Things is a huge place, with plenty of potential for a variety of sensor-friendly application protocols. The essential goals, designs, and capabilities of all protocols are different. It's critical to recognize the class of used addresses by these protocols. Before selecting a protocol for a specific application, the major system

https://doi.org/10.33640/2405-609X.3165

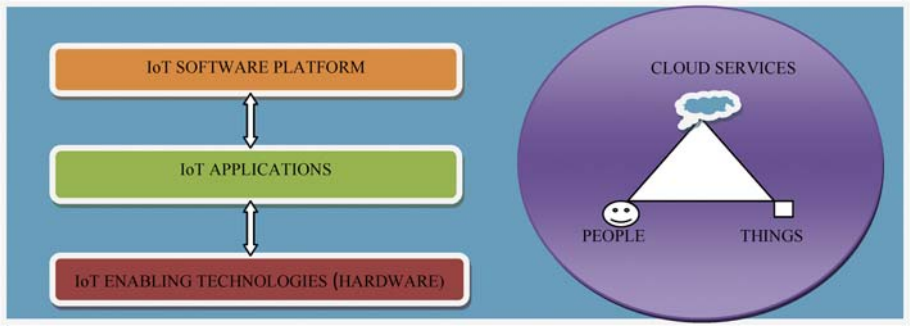

Fig. 1. IoT architecture [1].

needs like QoS, performance, interoperability, security, and fault tolerance are taken into account. To alleviate the issues of testing remote components, protocols such as a ZeroMQ-based simulation framework, for simulating distributed components, have been developed [10].

Communication protocols play a major role in safeguarding the network. The internet carries on at a rapid pace, as does the computer network. The Internet brings with it an increase in cybercrime in networks. As a result, understanding the protocols that control data flow in a network is critical. As a result, the designing of secure protocols becomes more important. Considering the study of [11], the design goals for safety-critical IoT are shown in Fig. 3.

Our purpose in this study is to analyze differences in communication protocols utilized by the server. It is considered that a limited number of IoT devices send data from sensors over the Internet. So, to achieve high-quality communication in a large-scale network for real-time data collection, protocols ought to include elements such as communication overhead, security, throughput, bandwidth, and support to QoS. This study

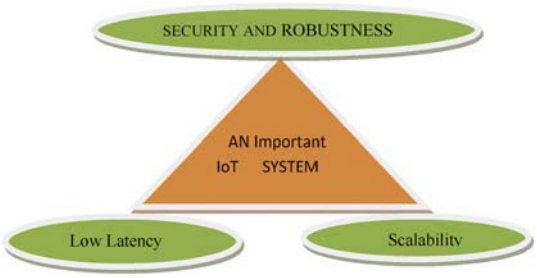

Fig. 3. IoT design goals for safety-critical IoT [11].

encompasses emerging protocols. Therefore, the study presents thoroughly analyzed results of protocols. Our results allow readers to see the flaws and pros of communication protocols used between IoT devices and servers.

2. Method

A comparative analysis has been done among the protocols based on 6 parameters to check which protocol stands out among the others. Communication

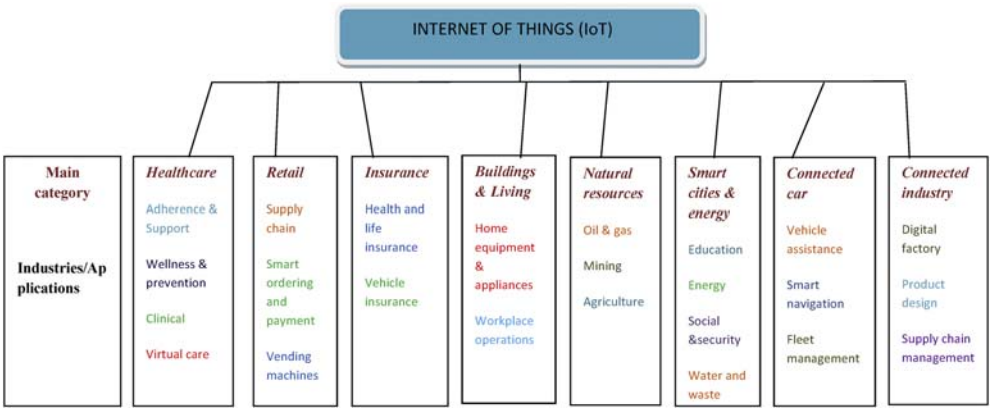

Fig. 2. IoT analytics [9].

protocols that can effectively manage complex situations are required for IoT implementation [12]. The 6 parameters, for the star rating, are listed below:

- Communication overhead
- Security
- Packet Loss
- Throughput
- Bandwidth
- Support to QoS.

The reasons for selecting the above parameters are stated as below:

2.1. Communication overhead

The strict requirements for low-latency communications are a typical feature of various applications. For a given typical modest payload size of IoT applications, reducing the overhead message size is crucial. The overhead message size includes pilot symbols for channel estimate, identification information, and control data. Low-overhead communications also contribute to the increased energy efficacy of IoT devices [13]. So, we have taken communication overhead as our first parameter.

2.2. Security

IoT setup comprises assigning distinctive identities to machines. IoT allows devices to communicate with one another through the web via public or private networks [14]. IoT devices are made up of actuators, wireless sensors, and computing devices. The devices allow data to be transmitted automatically among people or objects without any human intervention. Many IoT devices are unsecured and impossible to access on a physical level. Energy, cost, power, and lifetime are all factors that IoT devices must meet, but the most difficult need is security [15]. Hence security becomes our second important parameter in the analysis.

2.3. Packet loss

Small units of data known as packets are dispatched and accepted when accessing the web or some other network. Packet loss occurs when at least one of these packets does not arrive at its planned location. Sometimes the supplied data is never received, hence cannot be counted as throughput. Packet loss affects throughput for a given sender. Packet loss lowers throughput indirectly because certain transport layer protocols interpret loss as a sign of congestion and alter their transmission rate accordingly to prevent congestion. Assuming no retransmission, packets with the longest delays may be dropped first following a decrease in total latency. Packet loss affects users in the form of network disruption, delayed service, or complete network connectivity loss. Therefore packet loss becomes our third important parameter of consideration [11].

2.4. Throughput

A practical measure of actual packet delivery is data throughput. Packet arrival is critical to a network's high-performance service. People expect their demands to be heard and answered promptly when they use programs or software. Low throughput implies difficulty such as packet loss. Packet loss leads to poor or slow network performance. When it comes to troubleshooting, using throughput to assess network speed is beneficial. Throughput may pinpoint the real reason of a slow network. It also alerts managers to issue such as packet loss. Throughput is a more essential metric of network performance. Throughput tells us whether the network is actually slow or just pretending to be slow. By looking at the average data throughput, the user may observe how many packets are arriving at their destination. Packets must effectively reach their destination to provide a high-performance service. If a large number of packets are lost in transit the network's performance will suffer. Monitoring network throughput is critical for enterprises. The enterprises want to keep track of their network's real-time performance to ensure packet delivery, hence making it our fourth important parameter [16].

2.5. Bandwidth

Bandwidth means the part of data that may be sent and asynchronously received. The highest transfer throughput capacity of a network is known as network bandwidth. It's a metric for how much data may be transferred and gathered in a given amount of time. The bandwidth of a network defines how much data it can transmit and receive. The term bandwidth describes a system's capacity in place of its speed. It's vital to understand that bandwidth does not truly enhance a network's speed. Bandwidth makes the network appear to be quicker. Raising a network's bandwidth increases the amount of data that can be

delivered at once, not the speed at which that data is transmitted. The speed at which packets travel, is unaffected by bandwidth. It's also vital to keep in mind that greater bandwidth doesn't always imply high network performance. Large amounts of bandwidth will be useless if data performance is hampered by delay, jitter, or packet loss. So we have selected it as our fifth parameter. The essential thing to understand regarding bandwidth is that it does not ensure great network performance [17].

2.6. Quality of service

QoS is a set of technologies that operate together on a network. QoS ensures that high-priority applications and traffic it is reliably executed even when network capacity is restricted. QoS is the ability to govern network traffic-handling methods. The network satisfies the service needs of specific applications and users while adhering to network policies. So it becomes our sixth parameter of evaluation [18].

Various research papers [19–32] based on certain communication protocols were picked up and analysis of every protocol based on these parameters was done to reach a result.

3. Literature survey

Authors in Ref. [19] have proposed LIDOR a lightweight multi-hop protocol that stables IoT devices to IoT devices communication. LIDOR gives secure and private transmission. LIDOR employs feedback from the beginning to end system to check and fix broken paths locally. LIDOR effectively mitigates various types of DoS attacks. Including the fact that wormhole-supported greyhole attacks are difficult to detect, LIDOR's route selection converges. The LIDOR protocol has five stages:

- packet creation
- verification of packets
- forwarding of packet
- receipt of packet and
- acknowledgment handling

The source produces a packet, which it then sends to its most trustworthy neighbor. It sends out a probabilistic broadcast to all of its neighbors, encouraging them to try new things if reliability is poor. Duplicates are discarded and receiving nodes check that the packet is associated with a particular flow. And, much like the source node, they make forwarding decisions. On the reverse direction, the destination verifies the packet's validity and responds with an acknowledgment. Both receiving nodes confirm their validity and update their neighbor's reliability ratings. A penalty is imposed on neighbors' who do not return an acknowledgment. LIDOR was touted as the first algorithm of its kind to demonstrate confluence in the face of attacks by DoS. A conjunction is critical for a scheme's non-convergent property which can be used to set off an alarm for triggered DoS. They conducted several tests. These tests proved LIDOR's ability to withstand replay and wormhole attacks. In terms of the packet transmission ratio, LIDOR surpassed the standard project by 91% under attack from a replay and 32% under attack from the wormhole. LIDOR decreases overhead by 35% in a favorable situation while remaining unchanged under attack. Even in greyhole attacks aided by wormholes, their experiments showed that LIDOR converges.

Castor was linked to four protocols from the literature by the authors in [20]. With no or minimal additional overhead, the protocol attains 2 times upgraded packet delivery rates. As illustrated in the paper, Castor is a flexible firm communication protocol. CASTOR is extremely tolerant to a broad variety of faults and attacks. They called a wireless ad hoc network with a restricted contact range made up of stationary or moving nodes. Here, nodes communicated with their neighbors directly over a wireless channel. Nodes did help each other to communicate over multiple connections. One and all nodes had a distinct identity. Nodes that adhered to the system protocols were regarded as right. Nodes that do not were regarded as adversarial. PKTs and ACKs were the two types of messages used by Castor. The fields of the ACKs and PKTs assured 2 important properties to make sure the right routing state updates. First, an intermediate node obtained an ACK whether the target has obtained the corresponding PKT. And the second condition was no other node than the flow H's source was allowed to produce a PKT which can be authenticated as H's property since ek was an encrypted ACK authenticator. Ultimately, M denoted the payload that usually involved an integrity check. The ACK contains 1 sector, ak, that was utilized to verify that the corresponding PKT has been transported to target. In broadcast packets, Castor contained the data payload. However, the returned bandwidth cost remained the same or lower.

Authors in Ref. [21] demonstrated that Xcastor executes equivalently well as Castor in terms of reliability. Xcastor serves many groups simultaneously in comparison to Castor which have significantly longer

delivery times. The tests were carried out in a mobile environment. On the network layer, the question of dependable group communication was discussed. They developed and introduced Xcastor. XCASTOR is the world's first safe precise multicast routing protocol. They expanded Castor unicast routing protocol's secure and scalable routing principle. Thus efficient and trustworthy transmission for a huge count of mini-groups was observed.

In the paper [22], Secure Multihop Device to Device (SEMUD) is designed to comply with 3GPP Proximity-based Services, the level for implementing Device to Device transmission. SEMUD includes three ProSe-specific components, namely ProSe Application, ProSe Feature, and ProSe Application Server. SEMUD is a reliable multi-hop Device to Device, a 5G mobile network solution. Also if there isn't any infrastructure to sustain it, SEMUD allows for reliable and stable communication. SEMUD was implemented in the ns-3 simulator. The authors demonstrated its resistance to strong adversaries and attacks through comprehensive simulation.

For the Internet of Things, authors in Ref. [23] proposed a way out to secure Human to Thing (H2T) communications between CoAP-enabled sensors and HTTP hosts from end-to-end attacks. H2T interactions were the subject of the paper, which are needed in many important IoT applications. Technological and material heterogeneities characterize this communication style, making it exposed to DoS attacks thus making protection difficult. The paper discussed the issue of security in H2T communications. The solution was proposed with an asymmetric-selective mechanism to ensure efficient DoS safety. A selective safety mechanism was also included. The proposed solution managed security-related preferences of CoAP reactions at the application level based on WSN specificities. The results of the evaluation revealed that the suggested asymmetric and selective protection approach effect on CoAP servers' reduced security overhead. This also significantly reduced the impact of DoS attacks. According to the findings of the assessment, the proposed security plan eliminates H2T contact delays.

For the IoT definition, authors in Ref. [17] dealt with the different data protocols, CoAP, XMPP, AMQP, MQTT- SN, MQTT, and DDS. The authors wanted to see each data protocol compared to the others in terms of performance metrics including latency, bandwidth usage, packet error rate, and message size. The efficiency of one and all protocols was assessed based on the application. Based on theoretical values and values obtained from Contiki software, the paper evaluated and analyzed the performance of various data protocols such as XMPP, MQTT, DDS, AMQP, and CoAP. They displayed the latency of various data protocols with varying bandwidths. When compared to other data protocols, the CoAP Protocol was found to have lower latency. DDS also had constant telemetry latency that varied depending on network bandwidth. Unlike MQTT, AMQP showed a decrease in latency as network bandwidth was increased. XMPP latency, on the other hand, increased until a certain point and then decreased as network bandwidth increased. Since there was no re-transmission, it was found that CoAP retained the same bandwidth throughout. In comparison to other data protocols, DDS used a huge amount of bandwidth. Because of their retransmission mechanism, the TCP-based protocols XMPP, AMQP, and MQTT used more bandwidth and had a higher packet loss rate. However, it was discovered that as the packet loss rate increases, bandwidth utilization decreases for all three data protocols. They checked all of the data protocols with clear packet loss ranging from 0% to 25%. The TCP-based XMPP, AMQP, and MQTT protocols were found to have different characteristics than the UDP-based CoAP and DDS protocols. However, the packet loss rates for CoAP and XMPP on the created network were quite similar. MQTT, AMQP, and DDS, on the other hand, do not suffer from packet loss due to topic queues.

MQTT, CoAP, WebSocket, and XMPP were among the communication protocols handled by authors in Ref. [24]. The paper's main goal was to evaluate protocol efficiency in inhibited devices to give efficient transmission. As a result, a clever parking situation was created to contrast protocol response times when diversified traffic loads. CoAP outperforms at lower server use, than other queue-based protocols, according to the findings. When the program can handle multi-threading, XMPP outperforms other protocols in terms of server use. Besides WebSocket, when server usage is expanded, the average response time of protocols also expands. Other protocols use FIFO scheduling, while the most open processes are served first by XMPP.

Authors in Ref. [25] suggested a two-level phantom routing protocol to correct a few of the flaws in a false source packet routing protocol that already exists in their report. To overcome the limitations, the proposed protocol substituted a second-level phantom node at random for bogus origins packets. The results of their study showed that the proposed protocol could provide

good Source Location Privacy (SLP) security while reducing communication overhead. The protocol reduced energy utilization, increased packet delivery ratio (PDR), and reduced end-to-end (EED) latency by removing bogus packet traffic from the network. The results of the study revealed that when the sink node was 60 hops from the source node, the proposed protocol's protection duration was 3.4 times longer than the PR protocol's. The protocol used phantom nodes that were not connected to the source node. The protocol used a network of long backbone routes with many detours. Fake packets were periodically emitted at the end of each diversionary path. As a consequence, the adversary who listens in was successfully blurred as well as a solid SLP defense was assured. Nonetheless, significantly high overhead was introduced. Twenty experiment scenarios were run in the energy usage, PDR, and EED performance review. The source nodes were set up in various parts of the WSN domain in each case. Each source node is connected to a sink node and 1000 packets were sent. After all of the packets had been sent to the sink node, the average amount of energy used by each sensor node was measured at various locations of a node for each scenario. The proposed protocol ensured that communication overhead was kept to a minimum. The protocol was able to effectively replace the source of fake packets. With a 2-level phantom routing strategy the protocol was able to reduce the count of transmitted packets in the network. As a consequence, sensor nodes consumed less energy. The traffic load was greatly reduced when fake packets were eliminated from the network. As a consequence, the sensor nodes used fewer resources.

The authors in Ref. [26] proposed an architecture for contact and firewalls that is fit for applications of IoT. The proposed architecture relies on local Certifying Authorities to decentralize authentication and authorization (CA). The local CA refers to a public-private asymmetric key pair in their model. By adding Auth as a local server, it offloads the computational burden of authorization and authentication apps connected to the internet of things. It cuts an IoT device's power consumption by a factor of a thousand. It is critical since IoT tools are limited in terms of energy, whereas Auth is not. The proposed firewall was successful in the face of DOS flood attacks. The protection offered by DTLS and the suggested architecture are equivalent. However, DTLS has a significantly higher resource requirement than the proposed solution. They also assumed that entities are known to Auth's information such as Name, IP address, and so on in their proposed solution. Future versions are planned to provide a Protocol for Auth's Discovery that can locate available Auth servers in a network infrastructure that already exists. It could include synchronization of time as well as hold up for publisher-subscriber protocols like MQTT.

Authors in Ref. [27] suggested a communication protection scheme. The Diffie-Hellman encryption algorithm was used for establishing communication among the nodes that were authenticated. AES and MD5 were used to encrypt and verify data transmission between nodes in IoT devices. The proposed approach centered on node authentication. The encrypted message was sent by the receiving node. This project included several stages. The stages led to the nodes' authentication as well as the protection of the message sent from the source node to the other node. Key-sharing was accomplished by creating private keys on the source as well as target nodes first, then public keys on the pair of nodes. The key which was to be shared was created using a private key of node A and the public key of node B at the source node. The key must be the same as created by using a private key of node B and the public key of node A. The message was hashed at the source node at the same time when node A generated its private key. The data that was to be transmitted was encrypted at the different nodes. Data integrity was maintained by using hash functions to verify the data to be transmitted.

The authors in Ref. [28] created a single-chip network platform. It enabled communication between the PC and the MCU. A liaison carrier was the Modbus protocol. The slave's "Master-slave" gadget is a single chip microprocessor. For completion of the program compilation, KeilC51 software was needed. The program was then automatically loaded into SCM using PZISP downloading software. Debugging was carried out using Modbus Master debugging software to complete the software platform building. The Modbus protocol is a serial communication protocol in and of itself. The serial port sends data through TX data lines and collects information over RX data lines. The goal of this paper was to create a reliable protocol control system. The system was formed upon a single chip microcomputer. The aim was also to give a simple platform for new applications. The protocol accomplished the desired purpose. The device has essentially accomplished the anticipated result after debugging and running.

The author in Ref. [29] evaluated the benefits of using the relatively new Node.js platform. They wanted to implement the DPWS specification in the form

of *Node.DPWS*. *Node.DPWS* is a simple and lightweight set of libraries which generates and deployes DPWS devices on heterogeneous systems with low resources. According to the results of the performance evaluation, *Node.DPWS* exceeded the most appealing alternative currently available, the WS4DJMEDS toolbox. The improved performance, scale, and development ease of *Node.DPWS* demonstrated that the DPWS-related tools now available to developers have a lot of potential for improvement. As a result, it was seen that it was worthwhile to continue working on the *Node.DPWS* implementation, to extend support to other WS* related protocols, starting with the WSSecurity specification, and to enrich its libraries with further capabilities. Finally, an effort is underway to create a website with comprehensive documentation to help the research and development community use and expand *Node.DPWS* (and the accompanying Node.js and DPWS technologies).

The authors in Ref. [30] presented the blueprint of three-port converters. This enabled easy transitions of 7 unique working modes, relying on the loads and sources. 2 suitable converter topologies had been recognized and chosen for future PV-battery systems investigation and design. Traditionally, mode transition was accomplished by using appropriate control algorithms and feedback signals to assign certain switching patterns. It resulted in a slowdown of replies and circuit noise which was unavoidable. In addition, 3 current sensors and 3 voltage sensors were typically needed in TPCs for mode selection conclusion making. It was anticipated that the sensor miscalculations may result in an erroneous answer. The research provided a new control approach for previously reported topologies. The approach reduced the number of switching patterns from a minimum of five to just three. As a result, decisions were simplified. The transition happened organically formed upon load demand and power availability. Furthermore, rather than 6 sensors, only 3 voltage sensors and 1 current sensor were needed to perform all of the required activities, including battery protection, output regulation, and MPPT. Furthermore, these sensors were not involved in the mode selection process, resulting in a smooth and quick mode changeover. Furthermore, this approach examined two bidirectional ports. The most described topologies consider only one bidirectional port. Both independent and DC grid-connected applications were possible with this arrangement. The proposed solution was supported by experimental data. In comparison to standard PID controllers, the adoption of an FLC controller has permitted the construction of a system with a high degree of precision. This accomplishment was made possible by a comprehensive formulation of the inference rules. It covered all possible scenarios and allowed for an increase in the accuracy of the reference speed coupling value. Furthermore, the proposed solution entailed a lower-cost implementation of the genuine system. The advances in energy efficiency were attributable to the use of fuzzy logic, which allowed for more precise and flexible error correction. It also resulted for a faster and more optimal circuit response, resulting in the lowest possible charge usage. Another benefit that was gained through the usage of the FLC is the elimination of purely mathematical modeling of the system. When the two systems were compared, this trait became apparent. The purely analytical description of the circuit components was used to depict the comparison model in the literature. The model given was based on fuzzy logic control, which did not necessitate a fully analytical formulation. However, there were certain drawbacks to using this soft computing technique, such as the difficulty in defining all of the inference rules required for data evaluation or the possibility of developing an inadequate knowledge base.

In this paper [31] authors showed how to convert 2nd order compensated inductive power transfer (IPT) converters to 3rd order compensated IPT converters by including a capacitor or inductor to the output or input side. The parameters that were meant to be met to get the desired output were specified. The system's susceptibility to various parameter variations is then investigated. The results of the analysis of sensitivity gave a useful layout of reference for choosing parameters for higher-order IPT systems to accomplish the requisite LIV and LIC functioning with lesser design parameters. Furthermore, the analysis decodes the extra output-side or input-side roles of capacitors and inductors in making the entire system slighter sensitive, and thus gives a quick comprehension of the option compensation circuits of higher-order for applications dealing with a huge range of compensation network changes input variations and transformer coupling. Higher-order compensation circuits for inductive power transfer converters that address parameter variation and expand the operating range for the applications in question are abundant in the literature. Higher-order circuits are widely considered to provide design freedom and greater flexibility by increasing the space of parameters. Variation of certain variables such as load fluctuation, transformer coupling variation, and input voltage variation must be combated in specific applications. Individual

topologies were put forward with thoroughly derived equations matching to the particular topology in much of the described work.

The design of 3 port converters (TPCs) for smooth transitions of seven various operating modes is shown in this study [32], which was dependent on the loads and sources scheduling. 2 suitable converter topologies were recognized and chosen for future PV-battery system investigation and design. Traditionally, mode transition was accomplished by using appropriate control algorithms and feedback signals to assign certain switching patterns. It resulted in a slowdown of unavoidable circuit noise and response. In addition, 3 current sensors and 3 voltage sensors were typically needed in TPCs for mode selection decision making, where sensor errors resulted in an erroneous answer. This research provided a new control approach for previously reported topologies that reduced the number of switching patterns from a minimum of five to just three. As a result, decisions were simplified such that the transition happens organically found on load demand and power availability, rather than being forced as in the traditional way. Furthermore, rather than 6 sensors, only 3 voltage sensors and a current sensor, were needed to perform all of the required activities, including output regulation, MPPT, and battery protection. Furthermore, such sensors were not involved in the mode selection process, resulting in a smooth and quick mode changeover. Furthermore, this approach examined two bidirectional ports, whereas most described topologies only considered one bidirectional port. Both independent and DC grid-connected applications are possible with this arrangement. The proposed solution was supported by experimental data.

4. Comparison parameters

This section seeks to provide a study guideline for choosing the right communication protocol. A comparative analysis of various communication protocols is given in Table 1. The differences between communication protocols are measured using a variety of metrics. The metrics includes communication overhead, security, packet loss, throughput, bandwidth, and support to QoS. 6 star rating is given where * means the lowest and ****** means the highest.

5. Results and discussions

After going through the literature survey we decided to compare 6 protocols from the various protocols read by us which are listed below:

Table 1
Comparison Table.

Protocol	Characteristics	Communication overhead	Security	Packet Loss	Throughput	Bandwidth	Support to QoS	Rating
MQTT [20,21]	Low power usage, M-M communication	The TCP connection is required which raises the overall overhead.	Yes	Handle lost packets on the Transport Layer	Fair	Low	QoS 0 - At most once (Fire-and-Forget), QoS 1 - At least once, QoS 2-Exactly [33]	****
XMPP [20,21]	Channel encryption and presence checking	High	Yes	Fair	Fair	Low	No	**
CoAP [20,21]	Synchronous request-response, 1-1 or M-M communication	Runs over UDP; connection overheads are not experienced. The header size is 4 bytes [33].	Yes	Higher probability of packet loss	Fair [34]	Low	Confirmable Message (similar to At most once) or Non-confirmable Message (similar to At least once)	****
CASTOR [20]	Each node with its own unique identity	Limited overhead	Yes	React to all causes of packet loss	Fair	Low	Yes	****
AMQP [17]	Message queuing and interoperable [17]	Highest. The header size is 8 bytes [33].	Yes	Fair	Fair	High	Settle Format (similar to At most once) or Unsettle Format (similar to At least once)	**
LIDOR [19]	end-to-end feedback mechanism	Very limited and Reduces it by 32% as compared to CASTOR [19]	Yes	No mechanism for lost packets	Excellent	Low	Yes	*****

a) MQTT
b) XMPP
c) CoAP
d) CASTOR
e) AMQP
f) LIDOR

The six protocols were compared on 6 parameters i.e. communication overhead, security, packet loss, throughput, bandwidth, and support to QoS. Based on the six parameters it was found that LIDOR performed the best among the six other protocols. It was an extremely outstanding communication protocol. It had very limited communication overhead. The security parameter was also satisfied. It had excellent throughput and low bandwidth. The support to QoS was also there. The only drawback associated with it was the unavailability of any mechanism for packets that were lost. Overall it got five stars out of six which was the highest in comparison to the others. It nearly topped in all the six parameters as it upgrades authenticity under DoS attacks approximately by 91%. LIDOR reduces network overhead by 32% as contrasted with CASTOR, which among the other protocols had limited overhead [19].

LIDOR performs cryptographic operations per-packet. LIDOR in a computationally efficient manner achieves end-to-end delays in the order of 1 ms in 5-hop testbed. This confirms that the computational overhead is insignificant [19]. The results of various parameters are presented by Figs. 4–11.

5.1. Communication overhead

Communication overhead incurred by AMQP was highest and by LIDOR was lowest. MQTT has the smallest message header size (2 bytes), but it requires a TCP connection, which adds to the overall overhead. The overall message size is also increased [35]. AMQP sends and receives messages in a variety of ways, including directly, in fan-out form, by subject, and formed on headers. AMQP is a protocol which is binary. AMQP needs an 8-byte fixed header and short message payloads. The message size varies depending on the server, programming technique or broker, [35]. The header size of CoAP is 4 bytes. CoAP runs on UDP due to which it doesn't experience connection overheads. It has been recorded that without any added overhead, Castor reliably achieves up to a 40% greater packet delivery rate [20]. In LIDOR the communication overhead is very limited and reduces by 32% as compared to CASTOR [19]. As LIDOR performs the best so the equations presented below show the overhead of LIDOR. In the flow, the 1st packet in LIDOR takes a nonce. The nonce is utilized by the destination node to re-build the Merkle tree. *l* hash values of the Merkle tree are needed by the intermediary nodes to confirm whether the packet is of a similar flow.

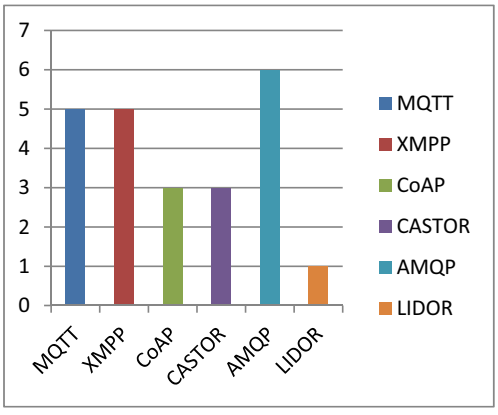

Fig. 4. Communication overhead incurred by different protocols.

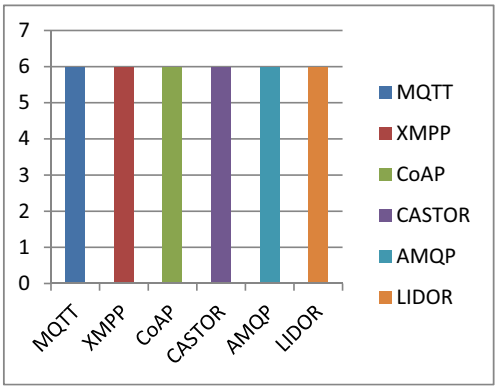

Fig. 5. Security ensured by protocol.

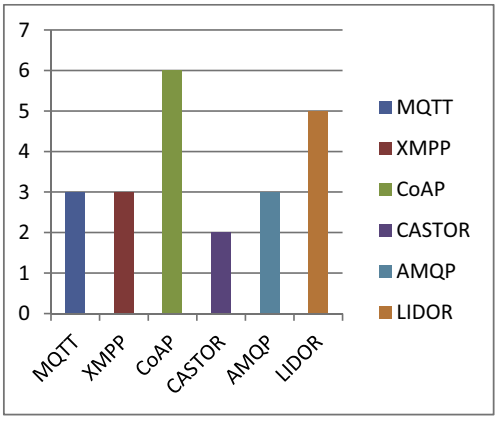

Fig. 6. Packet losses recovery mechanism in protocols.

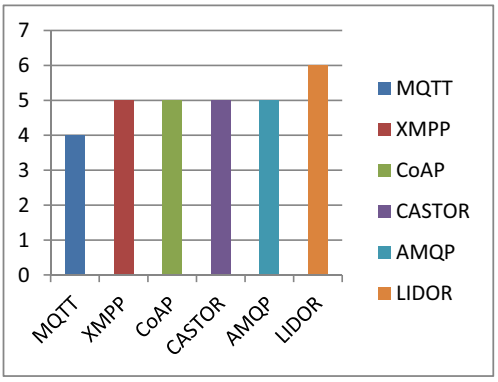

Fig. 7. Throughput of protocols.

Following [19], the overhead for the 1st packet (N_1^L), incurred by LIDOR can be given by equation (1):

$$N_1^L = Tl|hash(.)| + T|m| \quad (1)$$

where,

$|m|$ = size of the nonce in bytes
T = count of hops among the destination and source nodes.
$|hash(.)|$ = the size of 1 hash value in bytes.
(.) are unknown nodes. So the count of packets passed on to Merkle tree flow is given by 2^l.

If r denotes the hash values number needed to be transferred when the same node gets all the packets, where r is an integer that is non-negative in the range [0, l]. Then,

$$\sum_{k=1}^{2^l} r = 2^l - 1 \quad (2)$$

This means that when all packets are forwarded to the same node, the total count of hash values to be communicated is 2^l - *1*. The nonce is redelivered in future packets if $p_k + QTT > p_{k+1}$ where p_k is the sending time of the kth packet.

QTT = the round-trip time.

The scenario which is considered as the best-case is when the nonce is only sent for the 1st packet. The scenario which is considered as the worst-case scenario is when the nonce is sent for every packet in the flow. So, it is assumed that nonce is transmitted for $2 \le k \le \alpha_m$ packets, where $\alpha_m (\le 2^l - 1)$. As a result, the chance of retransmission of nonce is shown by p_m, is given as follows:

$$p_{m = \alpha m \,/\, 2l-1} \quad (3)$$

For $2 \le k \le 2^l - 1$, ${}_K^L N$ is given by equation (4) as follows:

$$N_K^L = T(r|hash(.)| + pm|m|) \quad (4)$$

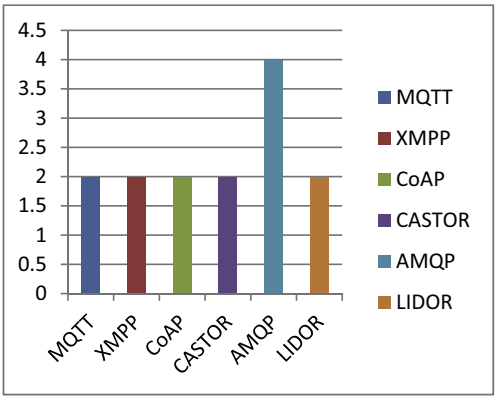

Fig. 8. Bandwidth of protocols.

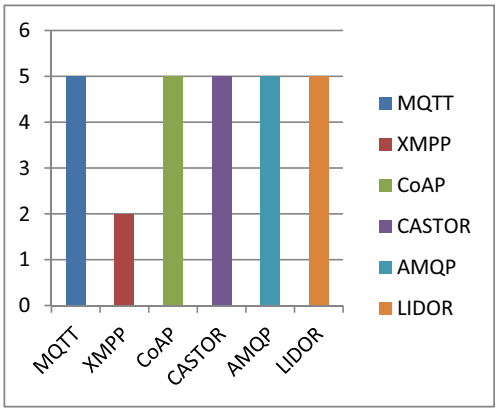

Fig. 9. Quality of service provided by protocols.

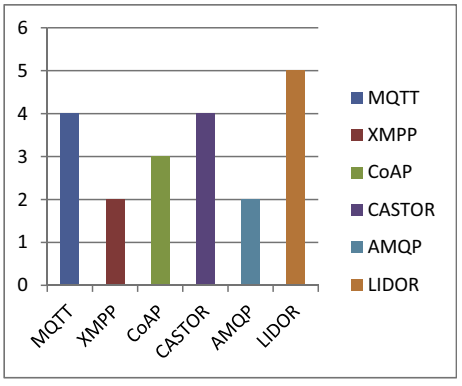

Fig. 10. The overall rating given to protocols.

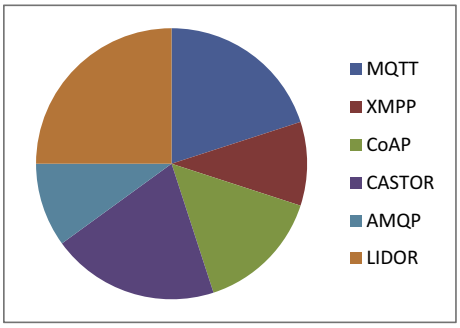

Fig. 11. Overall rating.

Using equations (1), (2) and (4), the overhead of LIDOR protocol (N^L) can be expressed using equation (4).

$$N^L = T((2l-1)|hash(.)| + (1 + (2l-1)pm)|m|) \quad (5)$$

5.2. Security

After comparing the various protocols with each other it was found that the security level given by all the protocols was almost of the same level. So every protocol was rated equally in terms of security**.**

5.3. Packet loss

The probability of highest packet loss was seen in CoAP. The probability of lowest packet loss was seen in CASTOR. CASTOR performed the best among the other protocols. CASTOR responded to all causes of packet loss, whether benign or hostile. On the Transport Layer, MQTT dealt with lost packets. Over the UDP format, CoAP provides datagram transport layer security.

It was learned that LIDOR too performed well. However in LIDOR once the packet was lost there was no mechanism to recover it. Now let us see how Castor which performed the best in our packet loss parameter is resilient to packet dropping attacks in the following explanation given below:

In CASTOR [20] a packet flow with id *P* from some correct source *1* to some correct destination *m* is considered. Nodes *1, 2, …,m−1*, are in charge of forwarding packets .

o = the adversarial node.

M does not receive a PKT dropped by o and does not respond with an ACK. CASTOR's acknowledgment method ensure that m is the only node capable of producing an ACK for the PKT. Every node *k = 1, …,o-1* preceding *o* on the route times out waiting for the ACK and lowers its reliability estimate S_{PK+1} for the successor on the route when the PKT is lost where, S_{PK} = an arithmetic average of two reliability estimators S^a_{PK} and S^f_{PK}.

The exponential averages of packet delivery rates are used in both reliability estimators. The lower the estimators become, the more aggressively *o* declines. One of the following happens for some *k = 1, …,o − 1: k* sends the packet to every neighbor or the reliability estimator S_{PK} of some neighbor $k' \neq k + 1$ of *k* exceeds S_{PK+1}, and *k* forwards subsequent packets to *k*. If some of *k's* neighbors are successful in delivering the PKT and receiving a proper ACK, *k's* reliability estimators are increased, and new routes are constructed. After another crowded drop, *k* eventually re-routes to *k,* away from the cause of unreliability. Castor's primary approach for removing lossy nodes from routes is through this mechanism. If the adversarial node *o* forwards PKTs but drops ACKs, the behavior is similar: nodes *k = 1, …,o-1* timeout waiting for the ACK, and the same procedure removes *o* off the routes. The only difference between dropping the PKT and receiving the ACK is that the ACK is received by the successors of *o* on the route, and their associated reliability estimators grow.

5.4. Throughput

The throughput of all the protocols was seen almost to be equal. However, LIDOR performed best among the others.

5.5. Bandwidth

MQTT protocol is a data-agnostic bandwidth standard with many levels of QoS support. Because all messages have a small code footprint, MQTT is considered a lightweight messaging protocol. However, XMPP is just desirable for short messages. AMQP is unreliable at low bandwidths, but as bandwidth increases, it becomes more dependable. LIDOR and CASTOR too show almost the same results.

5.6. Quality of service

AMQP provides dependable Quality of Services such as at least once, exactly once, and at-most-once. CoAP allows for multicast communication, which includes one-to-one and many-to-many communication.

XMPP does not provide QoS as it does not have explicit QoS control and rely on the underlying transport layer (e.g., TCP/IP) to check fundamental message dependability. Almost all the protocols except XMPP performed in the same manner here.

5.7. Overall rating

After comparing the various protocols on the six different parameters it was observed that LIDOR performed the best among the others. The rating of LIDOR was five out of six. MQTT, CoAP, and CASTOR performed equally well and got four stars out of six.

The worst performing protocols were XMPP and AMQP as both got 2 stars out of six. The results are further represented by a pie chart as shown below:

6. Research gaps and challenges

The following challenges and gaps were found and are stated as below:

- Various constraints of network overhead.
- Lack of physical protection.
- Another critical element of cloud protection is integrity. Violation and misappropriation of valuable data can be prevented by carefully using the resources. The admittance of an entity should also be done carefully [36].
- Traditionally devised security measures in the sense of ICT and networking technology become extremely difficult to enforce in control systems. Exchanging these legacy pieces is either prohibitively expensive due to sales loss during the transformation period or impossible [37].

7. Conclusions

Considering the related literature, this paper concludes that the furnishing of secure and robust transmission is very relevant for IoT applications. IoT applications require a high level of safety. The role of technology in our daily life is growing. The importance of communication in human life has made IoT-sanctioned systems a truth. Safety-critical IoT system has become an unsegregated part of our existence. In this study, MQTT, XMPP, CoAP, CASTOR, AMQP, and LIDOR have been studied, analyzed, and compared based on 6 parameters. The 6 parameters are communication overhead, security, packet loss, throughput, bandwidth, and support to QoS. For making it easier to understand the characteristics of the protocols are also explained. By comparing these characteristics we aim to reveal the differences of the protocols and bring out the most efficient protocol among the 6. As a result, LIDOR has been observed as the most successful communication protocol for the parameters taken into consideration. It has been seen that LIDOR has end to end feedback mechanism. The feedback mechanism helps LIDOR to reduce the overall communication overhead of the protocol. LIDOR upgrades authenticity under DoS attacks approximately by 91%. When contrasted with CASTOR, LIDOR decreases network overhead by 32%. CASTOR had limited overhead [19] among the other protocols. The throughput of LIDOR is excellent in comparison to others. The bandwidth is low. LIDOR shows high support to QoS. The only drawback is LIDOR does not respond to the packets which are lost once.

Conflicts of interest

None.

References

[1] I. Khajenasiri, A. Estebsari, M. Verhelst, G. Gielen, A review on internet of things solutions for intelligent energy control in buildings for smart city applications, Energy Proc. 111 (2017) 770–779, https://doi.org/10.1016/j.egypro.2017.03.239.

[2] H. Sundmaeker, P. Guillemin, P. Friess, S. Woelfflé, Vision and challenges for realising the Internet of Things, Cluster of European research projects on the internet of things, Publications Office of the European Union, Luxembourg, 2010. https://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.370.8561&rep=rep1&type=pdf.

[3] V. Karagiannis, P. Chatzimisios, F.V. Gallego, J. Alonso-Zarate, A survey on application layer protocols for the internet of things, Trans IoT Cloud Comput. 3 (2015) 11–17. http://icas-pub.org/ojs/index.php/ticc/article/view/47.

[4] B. Xie, A. Kumar, C.M. Chen, Future generation computer systems: Preface, Future Generat Comput Syst. 29 (2013) 1680–1681, https://doi.org/10.1016/j.future.2013.04.025.

[5] J. Zhang, H. Chen, L. Gong, J. Cao, Z. Gu, The current research of IoT security, in: 4th international conference on data science and cyberspace (DSC), IEEE, 2019, pp. 346–353, https://doi.org/10.1109/DSC.2019.00059.

[6] K. Wrona, Securing the internet of things: a military perspective, in: World forum internet things (WF-IoT), IEEE, 2015, pp. 502–507, https://doi.org/10.1109/WF-IoT.2015.7389105.

[7] F. Arena, G. Pau, An overview of big data analysis, Bull Electr Eng Inform. 9 (2020) 1646–1653, https://doi.org/10.11591/eei.v9i4.2359.

[8] J. Gubbi, R. Buyya, S. Marusic, M. Palaniswami, Internet of things (IoT): a vision, architectural elements, and future directions, Future Generat Comput Syst. 29 (2013) 1645–1660, https://doi.org/10.1016/j.future.2013.01.010.

[9] C. McLellan, The rise of industrial IoT, first ed., TechRepublic, 2019. http://book.itep.ru/depository/iot/SF_march2019_iiot.pdf.
[10] S. Lee, H. Park, W.J. Lee, Design of zero MQ-based cooperative simulation framework for distributed code and model components, Int J Futur Comput Commun. 4 (2015) 258–261, https://doi.org/10.7763/ijfcc.2015.v4.397.
[11] R. Marriswamy, R.H. Roogi, Packet loss in wireless networks, Int. J. Eng. Res. Tech. (IJERT). 3 (2014) 2881–2885.
[12] S. Al-Sarawi, M. Anbar, K. Alieyan, M. Alzubaidi, 8th international conference on information technology, Internet of Things Amman, Jordan, 2017, pp. 697–702.
[13] Y. Shi, J. Dong, J. Zhang, Low-overhead communications in IoT networks, first ed., Springer, Singapore, 2020 https://doi.org/10.1007/978-981-15-3870-4.
[14] D. Kim, Performance of UWB wireless telecommunication positioning for disaster relief communication environment securing, Sustain Times. 10 (2018) 1–11, https://doi.org/10.3390/su10113857.
[15] P. Mall, M.Z.A. Bhuiyan, R. Amin, A lightweight secure communication protocol for IoT devices using physically unclonable function, in: G. Wang, J. Feng, M. Bhuiyan, R. Lu (Eds.), Security, privacy, and anonymity in computation, communication, and storage. SpaCCS 2019, lecture notes in computer science, Springer, Cham, 2019, pp. 26–35, https://doi.org/10.1007/978-3-030-24907-6_3.
[16] T. Moraes, B. Nogueira, V. Lira, E. Tavares, Performance comparison of IoT communication protocols, in: IEEE international conference on systems, man and cybernetics, SMC), 2019, pp. 3249–3254, https://doi.org/10.1109/SMC.2019.8914552.
[17] M. Anusha, E. Suresh Babu, L. Sai Mahesh Reddy, A. Vamsi Krishna, B. Bhagyasree, Performance analysis of data protocols of internet of things: a qualitative review, Int J Pure Appl Math. 115 (2017) 37–47.
[18] X. Masip-Bruin, M. Yannuzzi, J. Domingo-Pascual, A. Fonte, M. Curado, E. Monteiro, et al., Research challenges in QoS routing, Comput Commun. 29 (2006) 563–581, https://doi.org/10.1016/j.comcom.2005.06.008.
[19] M. Stute, P. Agarwal, A. Kumar, A. Asadi, M. Hollick, LIDOR: a light weight DoS-resilient communication protocol for safety critical IoT systems, IEEE Inter Things J. 7 (2020) 6802–6816, https://doi.org/10.1109/JIOT.2020.2985044.
[20] W. Galuba, P. Papadimitratos, M. Poturalski, K. Aberer, Z. Despotovic, W.K. Castor, Scalable secure routing for adhoc networks, in: International conference INFOCOM, 2010, pp. 1–15, https://doi.org/10.1109/INFCOM.2010.5462115.
[21] M. Schmittner, M. Hollick, Xcastor: secure and scalable group communication in ad hoc networks, in: 17th international symposium a world wireless mobile multimedia networks, WoWMoM), 2016, pp. 134–146, https://doi.org/10.1109/WoWMoM.2016.7523512.
[22] M. Schmittner, A. Asadi, M. Hollick, SEMUD: secure multihop device-to-device communication for 5G public safety networks, in: International network conference, 2017, pp. 1–9, https://doi.org/10.23919/IFIPNetworking.2017.8264846.
[23] S. Sahraoui, A. Bilami, Securing human-to-thing interactions in the internet of things with asymmetric and selective mechanism, Secur Priv. 1 (2018) 1–18, https://doi.org/10.1002/spy2.38.
[24] P. Kayal, H. Perros, Through a smart parking implementation, in: 20th conference innovations in clouds, Internet and Networks, 2020, pp. 331–336.
[25] L.C. Mutalemwa, M. Kang, S. Shin, Controlling the communication overhead of source location privacy protocols in multi-hop communication wireless networks, in: International conference on artificial intelligence and communication, 2020, pp. 55–59, https://doi.org/10.1109/ICAIIC48513.2020.9065284.
[26] N. Maheshwari, H. Dagale, Secure communication and firewall architecture for IoT applications, in: Proceedings of 10th international conference communications system networks, COMSNETS, 2018, pp. 328–335, https://doi.org/10.1109/COMSNETS.2018.8328215.
[27] K. Quist-Aphetsi, M.C. Xenya, Node to node secure data communication for IoT devices using Diffie-Hellman, AES, and MD5 cryptographic schemes, in: Proceedings of international conference cyber secure internet of things, ICSIoT), 2019, pp. 88–92, https://doi.org/10.1109/ICSIoT47925.2019.00022.
[28] S. Ying, H. Jin, X. Wu, J. Chunjie, G. Wei, W. Ping, Research on modbus protocol implementation technology based on single chip microcomputer, in: Proceedings of 3rd international conference intelligent system engineering, 2019, pp. 127–131, https://doi.org/10.1109/ICISE.2018.00031.
[29] K. Fysarakis, D. Mylonakis, C. Manifavas, I. Papaefstathiou, Node. DPWS: high performance and scalable web services for the IoT vol. 33, 2015, pp. 60–67, https://doi.org/10.1109/MS.2015.155, 3.
[30] M. Giliberto, F. Arena, G. Pau, A fuzzy-based solution for optimized management of energy consumption in e-bikes, J Wire Mob Netw Ubiquit Comput Depend Appl 10 (2019) 45–64, https://doi.org/10.22667/JOWUA.2019.09.30.045.
[31] Y.C. Liu, J. Zhang, C.K. Tse, C. Zhu, S.C. Wong, General pathways to higher order compensation circuits for IPT converters via sensitivity analysis, IEEE Trans Power Electron 36 (2021) 9897–9906, https://doi.org/10.1109/TPEL.2021.3062228, 9.
[32] H. Aljarajreh, D.D.C. Lu, Y.P. Siwakoti, R.P. Aguilera, C.K. Tse, A method of seamless transitions between different operating modes for three-port DC-DC converters, IEEE Access 9 (2021) 59184–59195, https://doi.org/10.1109/ACCESS.2021.3073948.
[33] J. Sidna, B. Amine, N. Abdallah, H. El Alami, Analysis and evaluation of communication protocols for IoT applications, in: Proceedings of ACM international conference, 2020, pp. 257–262, https://doi.org/10.1145/3419604.3419754.
[34] V. Sarafov, J. Seeger, Comparison of IoT data protocol overhead, in: Proceedings of seminar on future of internet, munich, 2018, pp. 7–14, https://doi.org/10.2313/NET-2018-03-1.
[35] N. Naik, Choice of effective messaging protocols for IoT systems: MQTT, CoAP, AMQP and HTTP, in: IEEE international symposium on systems engineering (ISSE), 2017, pp. 1–7. http://ieeexplore.ieee.org/document/8088251/.
[36] N. Malik, D. Puthal, P. Nanda, An overview of security challenges in vehicular ad-hoc networks, in: Proceedings of international conference on inf. Technologies ICIT), 2018, pp. 208–213, https://doi.org/10.1109/ICIT.2017.14.
[37] A. Volkova, M. Niedermeier, R. Basmadjian, H.D. Meer, Security challenges in control network protocols: a survey, IEEE Commun Surv Tutorials. 21 (2019) 619–639, https://doi.org/10.1109/COMST.2018.2872114.