JOURNAL OF NORTHEASTERN UNIVERSITY

Volume 25 Issue 04, 2022 ISSN: 1005-3026 https://dbdxxb.cn/ Original Research Paper

AN ENERGY-EFFICIENT AND RELIABLE NETWORK FOR FOG COMPUTING AND INTERNET OF THINGS BASED ON RPL PROTOCOL

Satyam Shrivastava, Dr. Abhay Kothari

PhD Scholar SAGE University, Indore, Professor SAGE University, Indore

satyamshrivastava89@gmail.com, abhaykothari333@gmail.com

Abstract— With the increasing advancement in the applications of the Internet of Things (IoT), the integrated Cloud Computing (CC) faces numerous threats such as performance, security, latency, and network breakdown. With the discovery of Fog Computing these issues are addressed by taking CC nearer to the Internet of Things (IoT). The key functionality of the fog is to provide the data generated by the IoT devices near the edge. Processing of the data and data storage is done locally at the fog node rather than moving the information to the cloud server. In comparison with the cloud, Fog Computing delivers services with high quality and quick response time. Hence, Fog Computing might be the optimal option to allow the Internet of Things to deliver an efficient and highly secured service to numerous IoT clients In this article, we talk about the most important parts of RPL applications for the Internet of Things. In the past few years, advances in sensing and communication technologies have led to a rapid growth in the number of ways the Internet of Things can be used. This is possible because of how quickly different Internet of Things devices are being made (IoT). The Internet of Things is made up of devices that work together to form their own network architecture. In this architecture, each device has a limited amount of battery power, and the link isn't very reliable. This kind of network is sometimes called a "low-power and lossy network." In this paper, we describe a routing protocol that works well for networks with low power and high loss. The proposed protocol adds a new rank value so that the source node can send packets to the destination node using an appropriate destination-oriented directed acyclic network. This makes it easier for the source node and the destination node to talk to each other. The main thing that goes into figuring out the proposed rank value is the number of transmissions that are expected. We also use the amount of energy left over to decide which node should act as a relay for the packet on its way to its final destination. We ran simulations to test performance, and the results show that the suggested routing protocol increases the number of packets that are delivered. This was especially true in places where the bit error rate was high. Compared to the strategy that had been used before, the results showed that our method successfully keeps the amount of energy used by all nodes at the same level. In this paper, we newly propose a linear IoT model to deploy processes and data to devices, fog nodes, and servers in IoT so that the total electric energy consumption of nodes can be reduced

Keywords: fog computing internet of things; IoT; RPL routing protocol; network lifetime optimization; energy load balancing; ELB; performance evaluation.

I INTRODUCTION

Many industries and individuals are gradually becoming reliant on intelligent devices and desktops to deal with the day- to-day task. These intelligent systems generate information

Submitted: 08/12/2022 **Accepted**: 27/12/2022

through different applications and sensors. As an outcome, industries are producing and storing large volumes of information consistently [1]. The information generated from different types of sensors is on the rise after the evolution of IoT. With this rapid rise in the size of the information being generated and lack of ability of predictable databases to handle different forms of organized and unorganized information, big data analytics has got inordinate consideration at present. The data collected from different devices are being analyzed by various organizations to extract suitable understanding to take crucial decisions. At present; various industries need a powerful cloud-based infrastructure because everything is getting migrated to the cloud as it has different features offering pay-per-use, scalability, and accessibility. The prevailing cloud service offered by CC is Infrastructure as a Service (IaaS), Platform as a Service (PaaS), and Software as a Service (SaaS). All these cloud services are heading in the direction of everything as a Service (XaaS) [2]. Nonetheless, information produced from these millions of sensors, specified as Big Data cannot be processed and moved to cloud completely as this may majorly result in latency Furthermore, few applications of IoT require quick processing than CC present capacity. This issue can be resolved with Fog Computing which ties together the smart devices processing power situated close to the client to assist the usage of networking, processing, and storage near the edge [3].

The functionality of fog with IoT is to cut down the information transfer to CC for storage, analysis, processing, efficiency, and to improve the performance. Thus, the information gathered by the sensor devices is transmitted to network devices such as edge for temporary storage and processing rather than transmitting them to the cloud, thus decreasing the latency and network traffic [4]. The unification of IoT with Fog Computing generates a unique prospect for services, named as Fog as a Service (FaaS), where multiple fog nodes are built by the service provider across various geographic locations and operate as an owner to various in-habitants from different vertical places. Every node in the fog manages storage, computation, and networking capabilities .Fog is completely a distributed computing approach, it does not entirely depend on any integrated component like CC [5]. The latency issue of CC can be overcome using fog by utilizing the unused resources of different devices near the client. Nevertheless, it depends on CC to do major tasks. Unlike CC, fog is a distributed computing approach where different devices near the clients use computing capabilities that have less-features but a good computing capacity with several cores. Therefore, several smart devices like network device management, switches, base stations, routers, smart- phones, etc. are installed with storage and computing power which can perform as fog computing devices. Because of diverse organization and global connectivity, various research problems linked to Fog Computing are evolved. The deployable environment of Fog Computing and its requirements are the key issues in the Fog Computing principle.

This is the reason; the computing schemes that are present in the Fog Computing domain are diverse. Hence, the query that comes in is: In which way Fog Computing will grab hold of novel challenges of failure handling and resource management in a diverse domain? Therefore, it is essential to examine the precise requirements of all the other interconnected features such as services, simulations, fault tolerance, hosting issues, and resource administration. Various

researches on Fog Computing have been carried out. Here, we put forward the focus and literature domains of these research efforts in brief. Various descriptions, applications, and problems are de- scribed by [7] related to models of Fog Computing. [6] reviewed the Internet of Everything and Fog domain with a unified view of the Internet of Everything and Fog computing. [8] Presented a Fog Computing environment from the display place viewpoints of clients and developers for applications in the smart city towards constructing a viable sensing infrastructure. A categorized frame- work of Fog Computing and cutting-edge technologies like storage, data processing, security, transmission, privacy protection, and resource governance was surveyed by [9].Fog, edge, and CC ecosystems concerning many dimensions of platform abstractions, application features, and system frame-work were presented by [10]. Fog frameworks and algorithms established on six diverse assessment principles namely flexibility, interoperability, diversification, federation, Quality of Service (QoS) administration, and adaptability are presented by [11]. A classification of Fog Computing conferring to the recognized challenges and its important aspects is presented by [12].

Wireless sensor networks, also called WSNs, are an important part of building and growing the Internet of Things (IoT). These networks make it possible for low-end devices with few resources to connect to the internet and give users access to services that could change their lives. The IEEE 802.15.4 standard, which is the basis of WSNs and an important part of the Internet of Things, is one of the main standards that support low power and lossy networks (LLNs). This standard not only lays the groundwork for efficient and cost-effective operation, but it also defines the physical and data-link levels of the network. The Internet Engineering Task Force (IETF) made the IPv6 low-power wireless personal area networks (6LoWPAN) so that devices with less power could be added to the Internet.

This protocol is an adaptation layer that lets sensor nodes use the Internet Protocol (IP) stack and connect to other devices on the network. Because of these adaption layers, these nodes can use routing protocols at the network layer. They also make end-to-end connections possible, which are needed for a large number of applications. Traditional routing protocols can't handle the huge number of new nodes that are being added to the Internet because of its rapid growth and the rise of the Internet of Things. RPL was made just for LLNs because of this, and it quickly became very popular among people who work in the research community. In this paper, we acknowledge the importance of RPL as the standard routing protocol of the Internet of Things (IoT). We also present, for the very first time, a comprehensive review of RPL and RPL-based protocols in the context of the IoT, along with technical insights and recommendations for these implementations. In this way of doing the review

RPL PROTOCOL- RPL is a remote vector protocol that uses the IEEE 802.15.4 standard and supports the 6LoWPAN adaptation layer. It was made for IPv6 low-power devices in particular. The routing over LLNs (RoLL) working group came up with the routing requirements for LLNs in general, taking into account the limited energy, processing, and memory resources. This was done so that a lot of nodes could talk to each other using either a peer-to-peer topology or an extended star topology.This protocol sets up a multi-hop hierarchical topology for the nodes. This lets each node send data to its parent node, which

then sends it up until it reaches the sink or gateway node. You can think of this topology as a tree. In the same way, the sink node can send a unicast message to a certain node in its network to talk directly with that node. It gives an operation framework that makes sure connectivity in directions, resilience, reliability, flexibility, and the ability to grow. RPL makes sure that data routing works well and efficiently for nodes with limited resources. Some of the most important things about RPL are how well its hierarchy works, how timers are used to reduce the number of control messages, and how flexible the goal function is.

II MOTIVATION

RPL is a routing protocol for IPv6 networks that have low power and high loss. RPL was made to be a simple network protocol that can be run on devices with limited resources in industrial, environmental, and civil settings. This was done to support the internet vision of objects made up of thousands of devices connected across multiple networks. This was done so that the internet could work better. The first version of the RPL routing protocol has some problems that need to be fixed before it can reach its full performance and durability potential. In RPL, the objective function takes care of figuring out how far away something is (OF). The only two implementations that have been agreed upon so far are OF0 and MRHOF. On the other hand, these OFs create network topologies in which the traffic load on the bottleneck nodes may be very uneven, and the number of control packets may also go up. This is a big worry for the current OFs specified in RPL because it hurts the network's performance and makes it last less long. In this situation, it is not possible to develop a topology called DODAGs in a way that uses as little energy as possible. Also, the RPL standard doesn't have a switch that can be used to switch between the preferred parents (PPs) so that the load is spread out evenly. So, it is possible that one PP is given to multiple children who are able to use up all of his energy, and the path fails because this PP was lost. This will eventually have a negative influence not only on the performance of the network but also on its age. RPL needs to be able to let a single node have more than one parent node, but it can only work with one master node. Most of the traffic goes through this preferred main node, and the other original nodes are just used as backups. Because of this, PPs have a problem with a bottleneck, and they are much more likely to use a lot of energy than the other nodes in the topology. It is important to cut down on the control packet overhead to save power in each node, which will in turn make the network last longer. Because of this, we need to address these concerns and try to improve the standard RPL routing protocol so that we can use less energy, reduce the control packet overhead, better balance the traffic load, and make the network last longer.

RPL Hierarchy- As the base of the topology, RPL makes a directed acyclic graph (DAG) that doesn't have any edges that lead to other parts of the hierarchy. This makes it impossible for the structure to form cycles. The sink node starts building the first DAG and, in the process, becomes the last DAG root. Then, other nodes in this DAG start building their own DAGs, which are aimed at the first node to make a DAG with a destination orientation (DODAG). RPL builds and keeps track of its hierarchy with the help of different control messages.

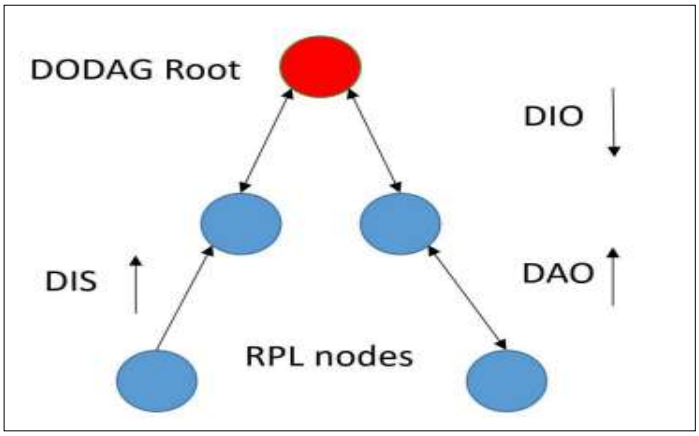

Figure 1. Control messages in RPL

The DODAG information object, also called the DIO, is sent out by the root node. This object has details about the sending node's rank, as well as its instance ID, version number, and DODAG-ID. This lets nodes keep useful information about the network that can help them make an informed decision. It also lets them choose whether or not they will do something when they get this message. The child node sends the destination advertisement object, or DAO, to the parent node, which is also called the DAG root or the DODAG root. This message tells the parent node where the message is going and, in effect, that the child node is still working. The root node could send a DAOack acknowledgement, but it doesn't have to. It is one of the most important things that RPL uses to keep connections going, and the DODAG information solicitation is one type of upward control message that is used to ask the parent node for a DIO. Figure 1 show how RPL control messages move from one place to another.

IoT and RPL Architecture - In a general sense, the architecture of the Internet of Things is composed of three layers, namely the Physical layer, the Network layer, and the Application layer. Sensors, RFID tags, smart meters, and several other detecting devices make up the Physical Layer, also known as the Sensing Layer. These sensors make measurements of physical characteristics like temperature, pressure, and humidity, and then report those measurements to the transport layer. The data is sent to higher layers after being processed in the Network Layer. The Network Layer is responsible for tasks such as the fragmentation of packets and the optimization of routing. The Data Management Layer of the Stack is responsible for storing and analyzing the collected data in order to obtain information that may be used. This layer implements cutting-edge methodologies like big data and cloud computing among others. The application layer is responsible for the design of application protocols as well as user interactions. The RPL-based Internet of Things architecture can also be broken down into three distinct layers. IEEE 802.15.4, IEEE 802.15.4 PHY, and 6LoWPAN, which act as an adaption layer, are responsible for managing the operations of the Data Link Layer and the Media Access Control Layer [13]. RPL, ICMP, and UDP are the protocols responsible for managing routing and transporting difficulties in the Network and Transport Layer. HTTP, CoAP, and MQTT are the three protocols that are designed at the Application Layer. The Internet of Things places demands on devices connected to the internet, including those for

low power consumption, low data transfer rates, limited memory capacity, and low processor capacity. The RPL architecture of the Internet of Things is broken down in Figure 2.

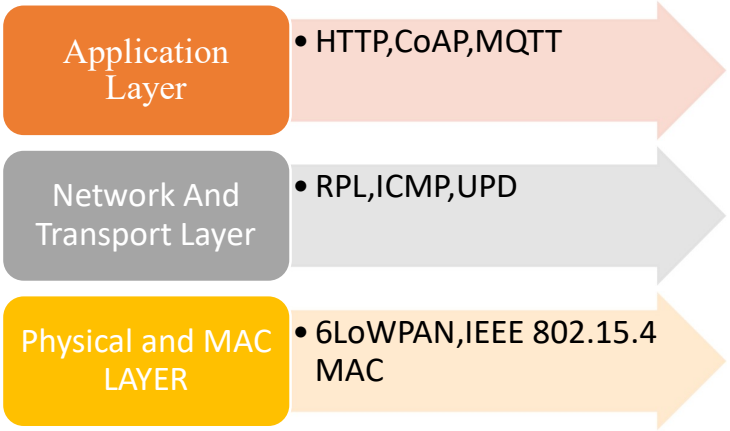

Figure 2 RPL architecture of IoT

III RELATED WORK

The term "Internet of Things" (IoT) refers to the internet's rapid growth and the ability to look at, collect, and share data that can be turned into knowledge or information. With the Internet of Things, the internet is no longer just a network of computers, but also of devices with different levels of similarity, like consumer electronics, home appliances, and wireless sensor networks (WSNs) [14] Also, the different information systems can work together and help support shared services [4]say that smart environments can be made in different places by setting up a direct link between different types of Internet of Things (IoT) devices, such as low power and lossy networks (LLNs). Because resources are limited in a big way, LLNs are fundamentally different from regular networks. For example, LLN routers have limits on how much processing power they can use, how long their batteries can last, and how much memory they can store. The lines aren't working as well because there are so many packet drops. These networks use a lot of different ways to talk to each other, including wired and wireless connections and many others. 6LoWPAN, which stands for IPv6 over low power wireless personal area networks, is a protocol that is a major step forward. It connects the world of IP to the world of low-power devices. This technology is based on IP and is made for low power wireless personal area networks (LoWPANs), such as wireless sensor networks (WSNs) that use IEEE 802.15.4 and IPv6 protocols. This brings a new dimension to the way 6LoWPANs are set up, allowing for the most interoperability possible through the use of the internet (Witwit and Idrees, 2018). The definition of 6LoWPAN says that routing is one of the most important problems that can happen in 6LoWPAN networks and should be looked into. So, the authors suggest three different kinds of multipath solutions based on RPL: energy load balancing (ELB), fast local repair (FLR), and combining the first two.

The work presented by [15] suggested Routing in LLNs is becoming more and more based on the RPL protocol. This protocol has a lot of extra work to do, and it doesn't spread the load evenly among the nodes in the parent list. Because of this, it doesn't work. The authors of El [16] came up with the idea of mobility enhanced RPL (MERPL) as a way to improve RPL and

make traffic along the routes more even. When it comes to the path's stability, MERPL does a lot better than regular RPL. But the authors don't think about a few important things, like the time between handoffs, how much it costs to send signals, and how much energy is lost in the process. The authors [17] suggest using EC-MRPL, which is a routing protocol that is both energy-efficient and aware of mobility (2017). This protocol makes sure that the nodes (MN) can always talk to each other so that access can be maintained no matter where they are.

The authors in [18] it was suggested that a new trustworthiness meter be added to the RPL's production and maintenance steps to make a better version of the RPL. This metric shows how much faith each node in the network has in the whole network. It is worked out by taking three things into account: selfishness, energy, and dependability. Thanks to this improved protocol, each node can decide whether or not it can trust the other nodes when building a topology. [19] Have made a good plan for balanced energy RPL architecture. They use the ELT scale, which is based on how well they can predict how long the bottlenecks will last, to build a DAG. Then, to make the most of what the DAG structure has to offer, they suggest using a multi-track strategy. The node uses all of its parents and adds to the weight of the traffic going through each of them. This makes sure that all nodes and bottlenecks use the same amount of power and that the power is shared fairly between them.

The work in [20] presented a new energy-based routing protocol (ER-RPL) in the area. This protocol allows data to be sent while reducing the amount of energy needed to do so without sacrificing dependability. Unlike traditional routing protocols, which depended on each node finding the path on its own, this one does not.

In [21] Most of the routing measures that are used now don't take into account how much energy the applications use. This causes the nodes to use different amounts of power, so the authors created more routing standards that use the power consumption of the contract as a guide. They relied on the RDC to give them an estimate of how much power the mother node used. For the calculation, they used power consumption and ETX. By using the suggested strategy, it was possible to improve the energy balance while keeping the beam delivery ratios and the energy efficiency. Zhao et al. came up with a routing protocol called HECRPL. It is based on RPL (2017). This protocol is based on hybrid cluster-parents that are both reliable and save energy. [22] Came up with a better version of RPL that they called "enhanced-RPL." In some places, the node can be spread out so that the problem of limited storage space caused by the node can be solved. Instead of only being on one parent, a subnet's prefixes are spread across several parents. The results of the simulation show that the suggested protocol beats the standard protocol by up to 30% for the number of packages delivered and by up to 64% for the amount of overhead. [23] Came up with the IRPL routing protocol, which is an improved version of the RPL routing protocol. This was done to keep the amount of energy used by WSNs stable. It is decided to use a topology control model to divide the space for communication into equal-sized rings. This clustering algorithm, along with a routing system, is the basis for the energy balance that happens.

IV PROPSOED SYSTEM

In addition to computers, various types of devices like sensors and actuators are interconnected in IoT (Internet of Things) [24]. Especially, a huge number of sensors are interconnected with things. In addition to requests from clients, large volume of data including multimedia data generated by sensors is transmitted to servers in networks. Processes to handle sensor data are performed on servers. Networks do not support enough bandwidth to transmit the sensor data with real-time constraints. In addition, servers are too overloaded to process and store the sensor data under time constraints the device layer is composed of sensors and actuators which are implemented in things. A sensor node collects data by events occurring in sensing physical environment and sends the sensor data to fog nodes named edge nodes. Data sensed by a server is forwarded to neighbor sensor nodes in wireless networks as discussed in wireless sensor networks (WSNs) [25]. Sensor data is finally delivered to edge nodes at the fog layer. Actuator nodes receive actions from edge nodes and perform the actions on the physical environment. Fog nodes are at a layer between the device and cloud layers. Fog nodes are interconnected with other fog nodes in networks. A fog node supports the routing function where messages are routed to destination nodes, i.e. servers and edge nodes. Fog nodes not only receive sensor data and forward the sensor data to fog nodes to deliver to server nodes. In addition, fog nodes do some computation on a collection of data sent by sensor nodes and other fog nodes. A fog node is also equipped with storages to buffer data. A fog node makes a decision on actions to be done by actuator nodes based on the data. Then, the fog node sends the actions to edge nodes and the edge nodes issue the actions to actuator nodes.

Model of fog nodes -There are two types of device nodes, i.e. sensor and actuator nodes. A sensor node just collects sensor data like temperature obtained by sensing physical environment and sends the sensor data to one or more than one edge node at the fog layer. On receipt of an action from an edge node, an actuator node performs the action on the physical environment. Fog nodes are interconnected with one another in networks. A fog node receives data from sensor nodes and other fog nodes, i.e. does the routing functions. Then, the data is processed, for example, an average value of a collection of data obtained from sensor nodes is calculated. Data processed by a fog node is sent to neighbor fog nodes and servers in a cloud. In addition, a fog node makes a decision on what action sensor nodes here to do based on sensor data collected from sensor nodes and fog nodes. In traditional cloud computing systems, every data sensed by sensor nodes is sent to servers in clouds. Then, the sensor data is processed and processed data is generated in servers. The sensor data and data obtained by processing the sensor data are stored in databases of servers. Then, servers send actions to actuator nodes in networks.

TX and Energy are two very important factors for figuring out how well OF is doing. Both ETX and energy-based OF have been looked at and judged by simulations. Figure 3 shows that the locations of the sender nodes are chosen at random in both of these situations. So that an evaluation can be done, the deployed area is split into two equal parts. The sink is on the 0 meter line, and the other sender nodes are spread out between the 0 meter line and the 100 meter line. Senders that are between 0 and 50 metres away from the sink are called "closer

nodes" because they are close to the sink. Far nodes are senders that are between 50 and 100 metres away. Fig. 2 and Fig. 3 show a comparison of the average amount of power used by ETX-based OF and Energy-based OF for nodes that are close to each other and nodes that are farther away. In RPL, each node's power usage can be broken down into four different types: I transmit power (also called Tx power), (ii) receive power (also called Rx power), (iii) CPU power, and (iv) low power mode (LPM) power. The average power used by each node in the energy-based OF is 1.56 mW, while the average power used by each node in the ETX-based OF is 1.291 mW. So, it shouldn't come as a surprise that the ETX OF has a much higher energy efficiency.

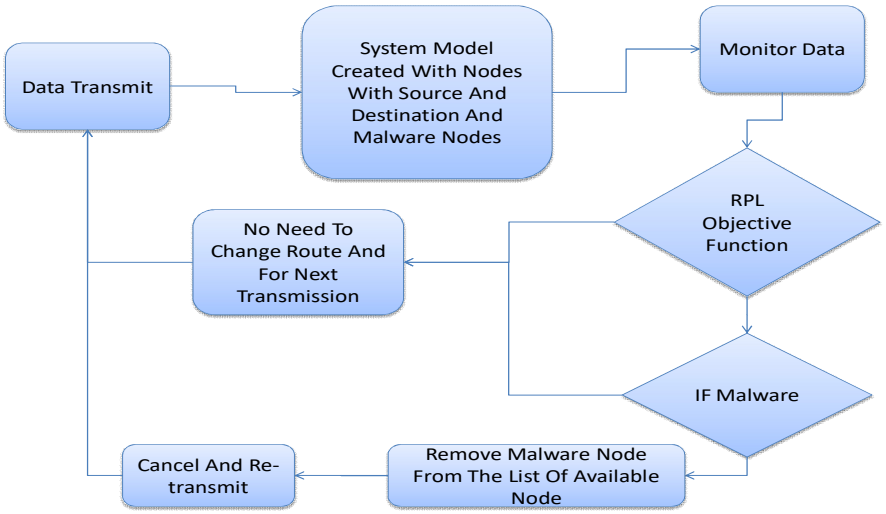

Figure.3 proposed block diagram

This article talked about a new routing protocol for the Internet of Things called energy-efficient load-balanced RPL (EL-RPL). In this protocol, an algorithm is given as a way to choose parents. It chooses a parent from the parent list to be the next hop node on the path to the destination node based on the highest amount of energy left and the total number of packets the parent has received. This could help make sure that everyone on the list gets a fair share of the work. The EL-RPL protocol also makes DODAG creation better by stopping DIO packets from being sent to nodes with lower ranks. This is how it helps to make DODAG better. This will lead to less energy use, which will in turn make the networks last longer. A lot of experiments were done with the MATLAB simulator in order to find out how well the RPL routing protocol works. The results show that the proposed RPL protocol can save energy, cut down on the number of control packets, and make IoT networks last longer than they do with other protocols that are already in use.

Figure 4 shows the simulation framework for evaluating multi-DODAGs in RPL. We used the MATLAB simulator to put our suggested method for evaluating the effectiveness of RPL-based sensor networks through its paces. The network's structure consists of 80 sender nodes and one root node for each DODAG. In Multi DODAGs, we have 80 sender nodes and 2, 3 root nodes. Table 1 shows what the Network Parameters mean and how to use them. We tested

the network by looking at single and multiple DODAGs for ETX (MRHOF), Hop Count (OF0), and Remaining Node Energy (RE). Once the results have been reached for network performance and reliability in terms of longer network life, they are written down and compared. The network topology is made up of nodes that are spread out over an area that is 800 meters wide and 800 meters long.

Network setup -ETX and Energy are two important ways to measure operating costs. Both ETX and energy-based OF have been looked at and judged by simulations. One sink node and nine sender nodes are used for the simulation. Figure 3 shows that the locations of the sender nodes are chosen at random in both of these situations. So that an evaluation can be done, the deployed area is split into two equal parts. On the line, you can find the sink at a distance of 0 metres and nine sender nodes anywhere from 0 metres to 800 metres away. Senders that are within the range are closer nodes because they are closer to the sink. The senders that are between 0 and 800 metres away are called "far nodes." Figures 7 and 8 show a comparison of how much power ETX-based OF and Energy-based OF use on average.

Objective Functions of RPL-RPL and Internet Communication Message Protocol are the two protocols that make up the network layer (ICMP). RPL is in charge of dealing with routing problems, and ICMP is in charge of dealing with communication messages. MP2P is what makes up most of the traffic that RPL can handle. Traffic from MP2P networks can be used in a wide range of collection-based applications. It can, however, also handle both point-to-point and point-to-multipoint (P2MP) traffic. The nodes in RPL use the Destination Oriented Directed Acyclic Graph (DODAG) to structure the network architecture. A scalar value called "rank" is used to decide where each node in the network should be placed. The DAG ROOT, also called the "sink," is the beginning of a never-ending rise in rank. DODAGID, which is a one-of-a-kind number, has been given to it.

Many DODAGs or sinks that are managed by the RPL network can be hosted by a single RPL instance. On the other hand, in one case, the DODAGs are optimized based on an Objective Function (OF), which can be found by an objective Code Point (OCP). OCP says that DODAG optimization should be used as either a constraint or a measure when building DODAG [6]. Metrics and constraints can be used to build a DODAG. Some examples are the number of hops, the latency, the expected number of transmissions, the energy of each node, and so on. DODAG Information Object (DIO) messages are sent by RPL nodes. These messages are used to build and maintain the DODAG. DAG ROOT gets started by sending multicast DIO messages to the nodes in the area. A DIO message has details about the RPL instance, the DODAGID, the number of the DAG version, the OF that was used, and the parent rank. All of this information is sent to the nearby nodes, which then update their own rank and send multicasts to the other nearby nodes. This process keeps going until a path from the leaf node to the root node is found by going through the hop nodes. If a new node wants to join the Multi DODAGs in RPL for Reliable Smart City IoT 73 DOG, it can either wait for DIO messages from neighboring nodes or send a DOADG Information Solicitation (DIS) to neighboring nodes. For both of these choices, the new node must wait for DIO messages from other nodes nearby. DODAG Acknowledgement Object (DAO) messages are sent when the path from the

root node to the leaf node needs to have P2MP communication. The way that one node talks to another is through P2P traffic [14-16]. Figure 4 shows how DODAG has grown over time. RPL functionality defines two types of routes: upward routes and downward routes. The direction in which data moves through a DODAG determines which route type to use. From the leaf nodes, there is a way up that can be taken to get to the DODAG root. The downward path leads from the root of the DODAG to the leaf nodes, while the chosen parent of a node is used to build the upward paths. When the node has information that needs to be sent to the root, it sends it right away to the parent that the node prefers. The message goes from the parent node to the parent of the parent node, and so on, until it gets to the DODAG root. In the downward route, the message is sent to the destination by the root. This can be done either by attaching the source route to the data packet or by sending the packet down the DODAG leaf node hop by hop [8]. RPL uses A trickle timer is used to reduce the number of control messages by only sending updates to the network when a problem is found. The trickle timer will send fewer control signals into the system as the network becomes more reliable. By using a trickle timer, you can save energy and cut down on the control traffic that comes with building and maintaining DODAG [17].

Djikstra's Shortest Path Algorithm

It is a generalization of the breadth-first search algorithm and can be seen as the ancestor of many path finding strategies used today.We start at the source vertex and move outward, giving each neighboring vertex the cost value of the edge that connects it to it.

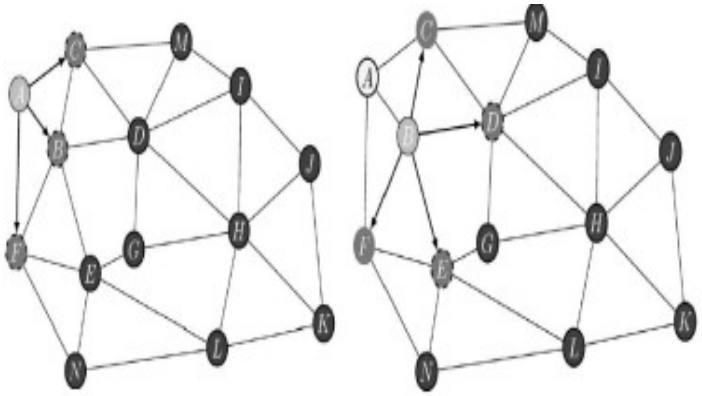

Figure.4 Djikstra's Shortest Path

Next, we'll look at the vertex that has accumulated the fewest costs so far, and we'll give that vertex's total cost plus the cost of each edge to each of the vertices that are next to it. This step is repeated as many times as needed until all of the processing on each vertex is done. If a vertex is looked at again, we will give it the new cost if it is less than the cost it already has. When the process is done, we will not only be able to find the single pair with the shortest path,

but we will also be able to figure out how far away any other vertex is from the source vertex .

Modified Dijkstra's Shortest Path Algorithm (MDSP)

The MDSP, or Modified Dijkstra's Shortest Path algorithm, is a brand-new idea for the shortest path algorithm. In this method, instead of using a single parameter, multiple parameters were used to find the legitimately shortest path. You can figure out how well the MDSP algorithm works in terms of the shortest path by measuring its nodes and then using the formula A1=randperm(N); to figure out how much time it takes.

Randomly Select Source Node

```
ind=A1(3)
pathL=[]
if(A==3)
  Source =1
  Dest=numel(X1)
  Rc1=Rc%4.
  matrizP(matrizP>Rc1)=inf
  [pathP,cost]=Ralgfun(Source,Dest,matrizP)
  costN=cost.*length(pathP)
  dist1L=costN
  dist2L=0
  apL=0
  if(~isempty(pathP))
   if(pathP(end)~=Dest)
      pathP=[pathP Dest]
   end
```

V SIMULATION RESULT AND DISCUSSION

In this investigation, the Dijkstra's shortest path Algorithm is used to find the shortest distances between source nodes A and a destination node B. Random data were used to figure out the weights of the edges in the graph that was made for the study. There is an introduction that sets the scene for a cost matrix that shows how long it takes to get from one node to another at different times of the day. This is something that needs to be done because the plan being proposed is meant to make it easier for people to move between cities in the same area. Dijkstra's algorithm can be used to find the shortest routes between any two points in a region. This can help you figure out how to get around the area. This will help you figure out what to do in the area. On the other hand, it has been found that using the route with the shortest distance between two specific nodes is not the best way to finish the shuttle service in the time allowed. This is because the best choice is not the route with the shortest distance between two given nodes. In this case, the values from the cost matrix are used to come up with a list of possible routes, taking into account how long it takes to get from one node to another at a certain time of day.

In this case, a customized version of Dijkstra's algorithm is used to figure out all the possible ways to get somewhere. The road system that was the focus of this study is shown by a weighted graph with forty nodes. The Internet of Things (IoT)[20-21] technology makes it possible for regular cities to become smart cities in a number of ways. When RPL is used to build the Internet of Things, the network becomes less stable and less efficient. This problem is what made people think of Multi DODAGs as a possible solution. When many DODAGs are used, data transfer can be sped up, control traffic can be cut down, and less power will be used overall. The fact that there are more nodes in multi DODAGs makes these graphs more trustworthy. In a smart city network, if a node or connection stops working, it could be very helpful to have an alternative path or sink for collecting point data. Also, multi DODAGs have places to collect data. These places are called "sinks," and they can cover a large geographical area.

A New Modified Dijkstra's Algorithm:

This algorithm assumes that traversing the shortest path is impractical.

Step 1 Use the conventional Dijkstra's algorithm to generate all the routes from a given source node i to the destination node j.

Step 2 Calculate the probability of traversing from i to j through all the nodes between i and j

Step 3 Output the result in descending order of probabilities

Step 4 Consider traversing the routes starting with the route with the highest probability

Step 5 Record the runtime of each route

Step 6 Output the runtime in order of efficiency

Step 7 Return.

Back-trace the shortest-path

```
r_path = [pathE]
i = parent(pathE)
while i~=pathS && i~=0
  r_path = [i r_path]
  i      = parent(i)
end
if i==pathS
  r_path = [i r_path]
else
  r_path = []
end
% Return cost
r_cost = distance(pathE)
```

Table 1 Simulation Parameter

Parameters	Parameters Value
No. of nodes	80

Area	800x800'
Node to CH power Ratio	0.045
Packet Size	2Mb/sec
no of cluster head	Nodes CH=10%
Energy per distance	0.001

An RPL-based Internet of Things network for smart cities can be made reliable by improving the network's performance and making it last longer. Performance metrics can be used to measure how well a network works. Some examples are the convergence time, the amount of power used, the amount of control traffic overhead, and the packet delivery ratio. On the other hand, the reliability of the network can be measured by the node participation metric. Our recommended Multi DODAG model aims to improve the RPLefficiency and reliability of smart city IoT.

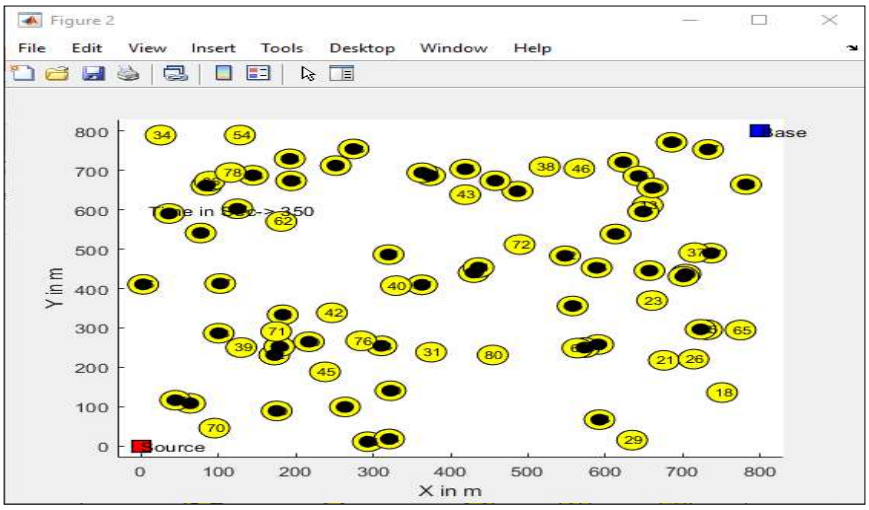

Figure.5 fog Node deployment

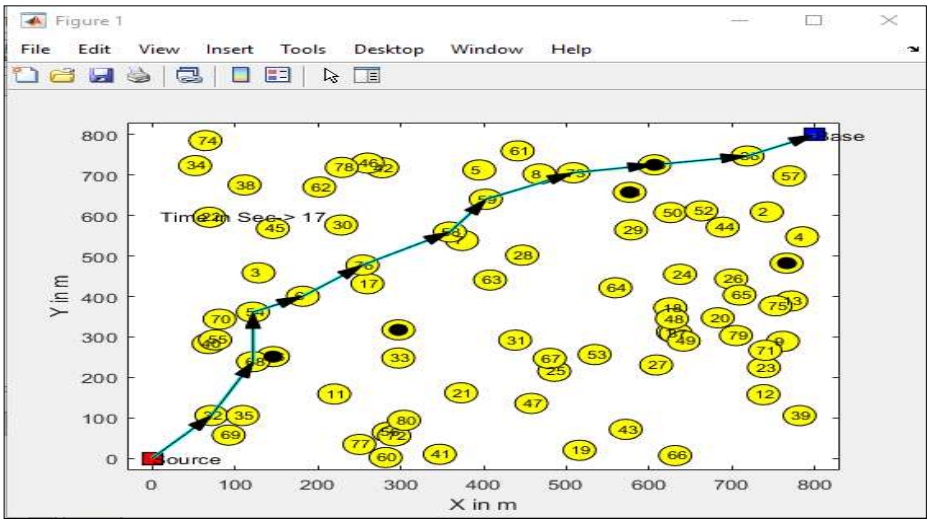

Figure.6 the sink/root initializes

Step 1 the sink/root will broadcast the first control messages DIO that contain the following information: the RPLInstanceID, DODAG identifier, version number, rank, and the OF supported by RPL that has been used to calculate the rank. After initializing the DODAG information, the sink/root will broadcast the first control messages DIO. Every node that is within the root communication range will be sent a DIO message, and it will be up to those nodes to decide whether or not to join the structure. The node that satisfies the conditions will determine whether or not the decision is made to add the node to the graph (if it has enough power to enter the DODAG construction process). The join is dependent on the node rank when the node satisfies the conditions; this rank is an incremental number that is calculated using the predefined goal function (OF). The node rank outside of the graph must be greater than or equal to the node rank within the graph for correlation to be valid. In this scenario, the DIO control message is disregarded because the node in question does not fulfill the requirements.

Fog Node Path

```
for i=1:noOfNode
  if transmat(pathS, i)~=inf
    distance(i) = transmat(pathS, i)
    parent(i)   = pathS
    queue       = [queue i]
  end
end
Width-first exploring the whole graph
while length(queue) ~= 0
  hopS  = queue(1)
  queue = queue(2:end)
  for hopE = 1:noOfNode
      if distance(hopE) > (distance(hopS) + transmat(hopS,hopE))
      distance(hopE) = distance(hopS) + transmat(hopS,hopE)
      parent(hopE)   = hopS
      queue          = [queue hopE]
    end
```

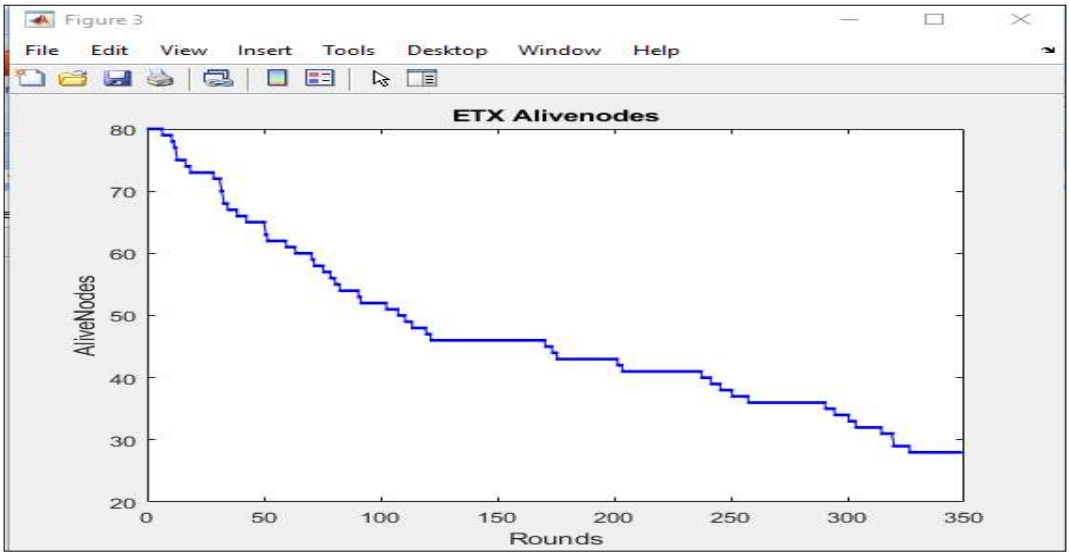

Figure.7 ETX alive fog node

Energy Function Algorithm

```
switch(A)
      case 1
       Direct
         Ec=beta.*dist2
      case 2
       LEACH
         Ec=alpha.*dist1 + beta.*dist2
      case 3
     Hopbyhop(custom)
         Ec=alpha.*dist1
      case 4
CORPS
         Ec=sum(sum(alpha.*dist1 + (beta.*dist2))
       case 5
     Proposed
         Ec=sum(sum(alpha.*dist1 + (beta.*dist2)))

      Otherwise
         Ec=0
  end
```

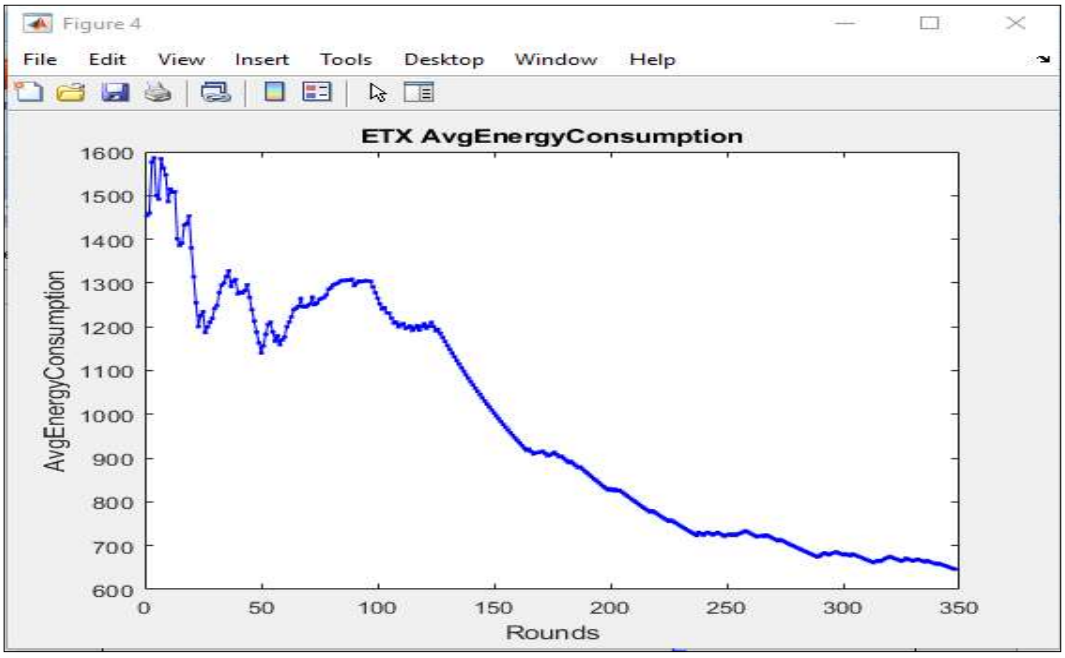

Figure.8 ETX Avg Energy Consumption

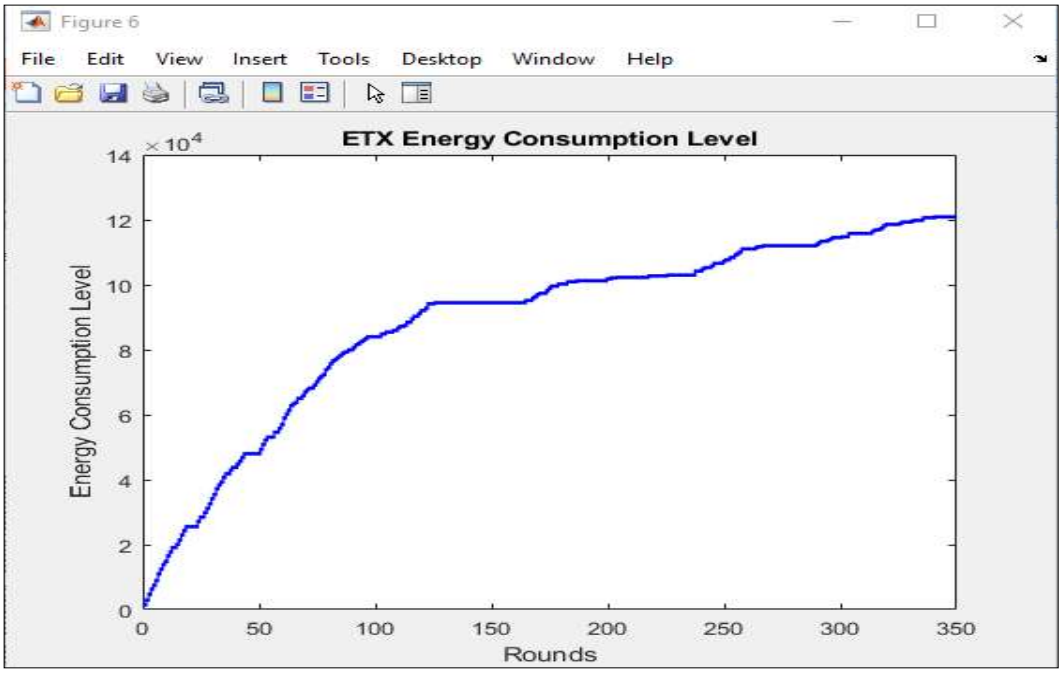

Figure.9 ETX Energy Consumption level

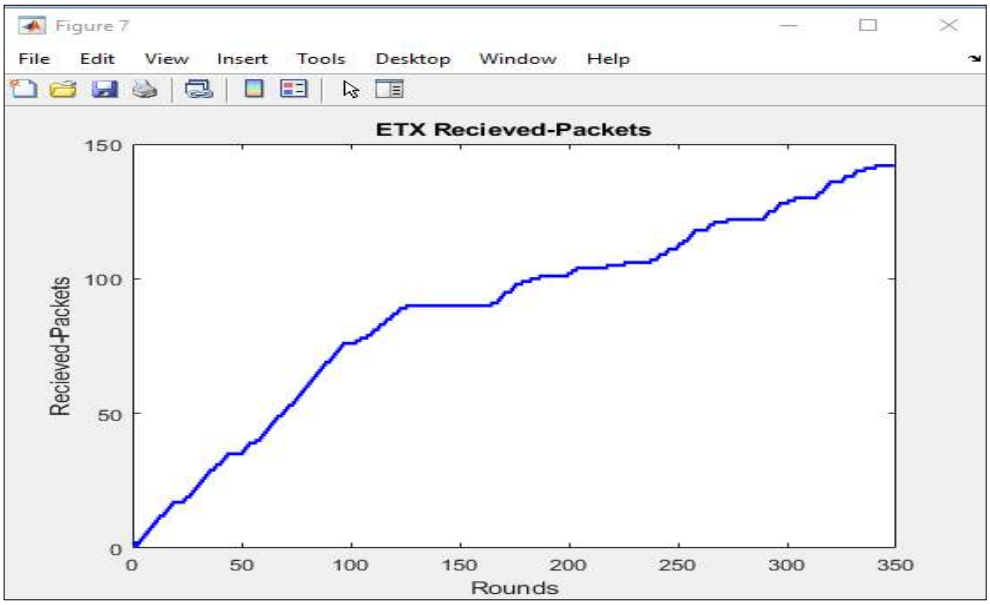

Figure.10 ETX Received-Packets

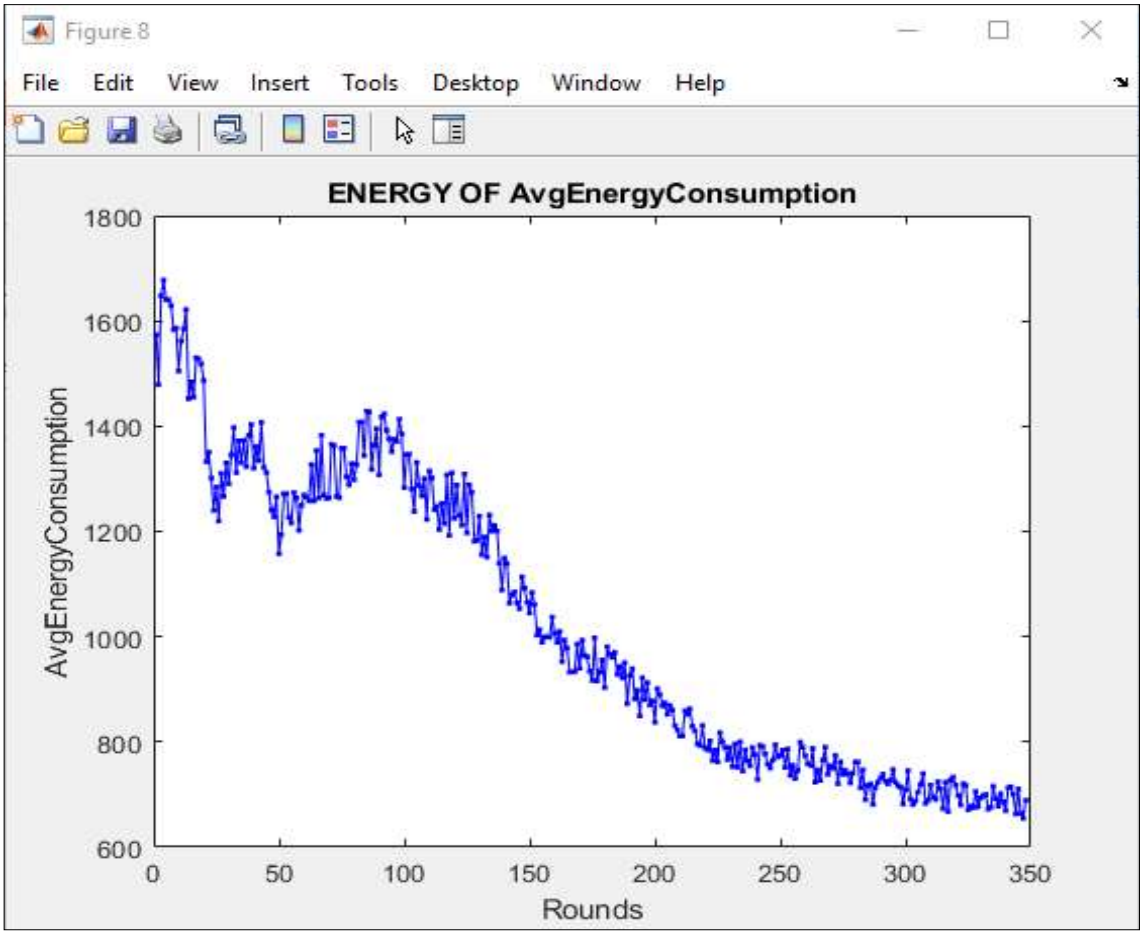

Figure.11 ETX OF Energy Consumption

Performance Metrics

We use the following metrics to capture the performance of our protocol and to compare it with standard RPL protocol.

Energy consumption In order to count the average cumulative energy consumption in the network, it is computed by the following formula:

	Proposed System Results	
	ETXOF	ENERGYOF
TXPower	7.179329	7.179329
RXPower	50.887787	52.688472
LPMPower	14.941470	13.922927
CPUPower	30.573747	33.847660

$$EC \leftarrow \left(1-\left(\frac{\sum_{j=1}^{P}\frac{\sum_{i=1}^{n}E_i}{n}}{P}\right)\right)*100$$

where EC is the average value of the energy consumption for each node during each period, n is the "number of nodes" in the network, P is the number of periods, and Ei is the total amount of energy that node I spent during all of those times.

Energy consumption

The amount of energy that is used has a big effect on how long the nodes will last. It has mostly to do with sending and receiving messages, processing on the central processing unit (CPU), and being idle or overhearing. Using the equation, you can figure out the average amount of energy each node used during each period.

Table 2 ETXOF and EnergyOF Proposed System

Table 3 ETXOF Proposed System and Existing System

	ETX OF	
	Proposed System	Existing System[1]
TXPower	7.17	7.99

RXPower	50.88	50.07
LPMPower	14.94	11.15
CPUPower	30.57	30.18

Table 4 ENERGYOF Proposed System and Existing System

	ENERGY OF	
	Proposed System	Existing System [1]
TXPower	7.17	12.17
RXPower	52.68	51.47
LPMPower	13.92	9.63
CPUPower	33.84	26.72

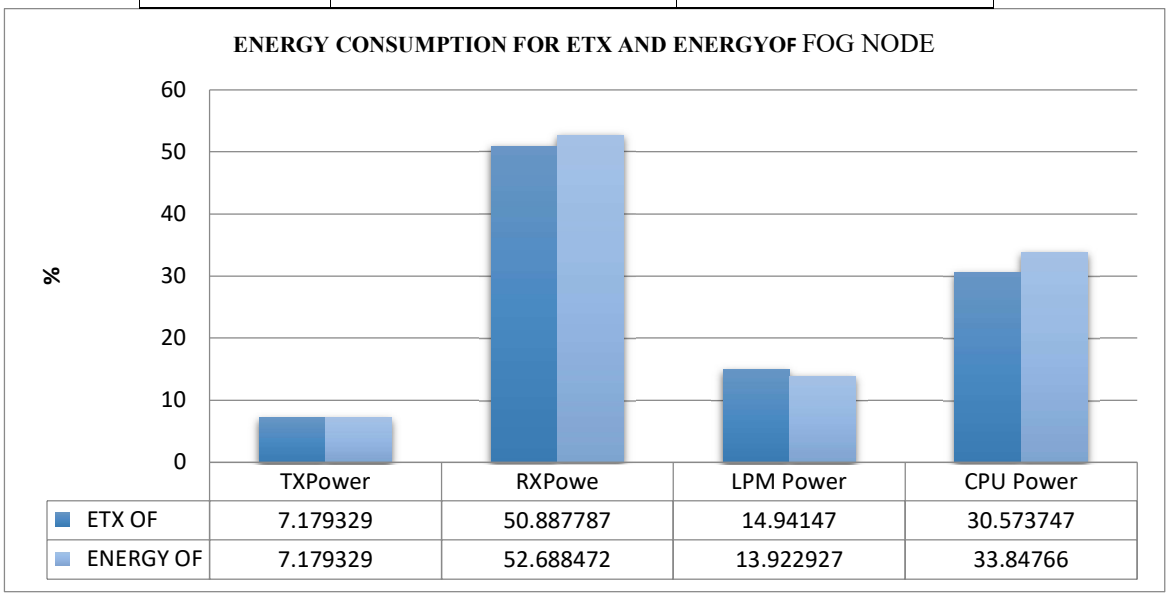

Figure.12 ENERGY OF Proposed System and Existing System

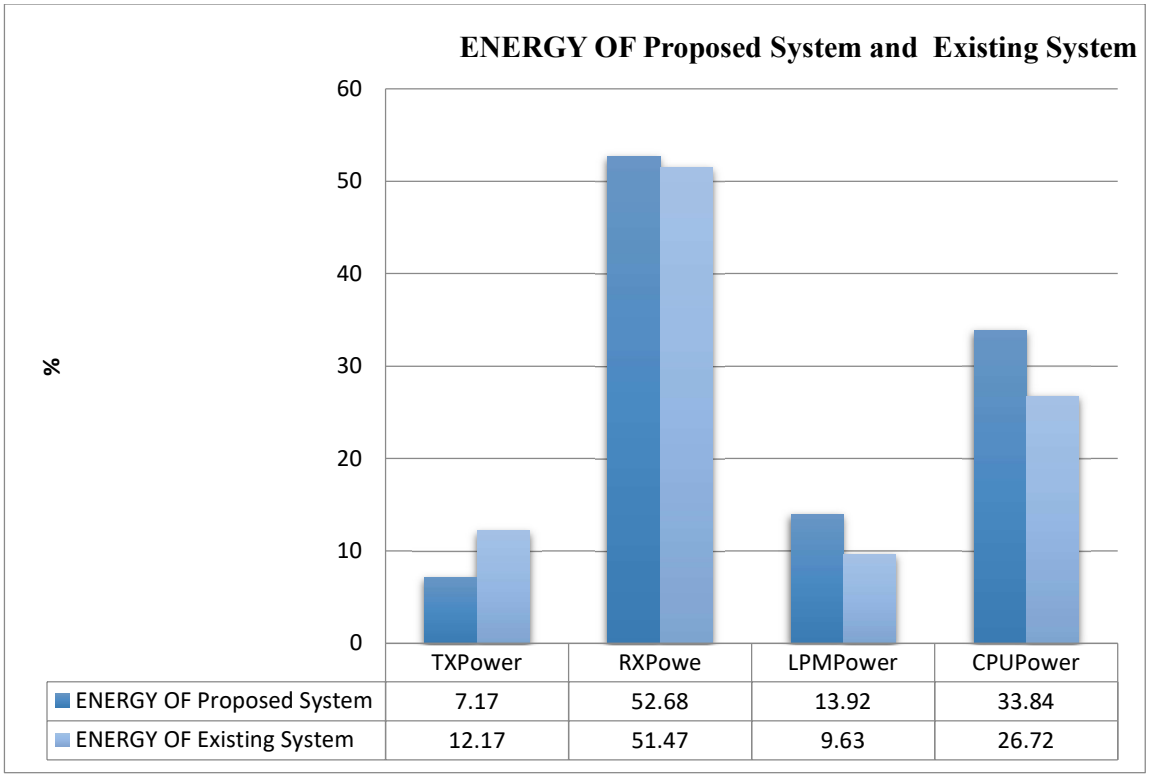

Figure.13 ENERGY OF Proposed System and Existing System

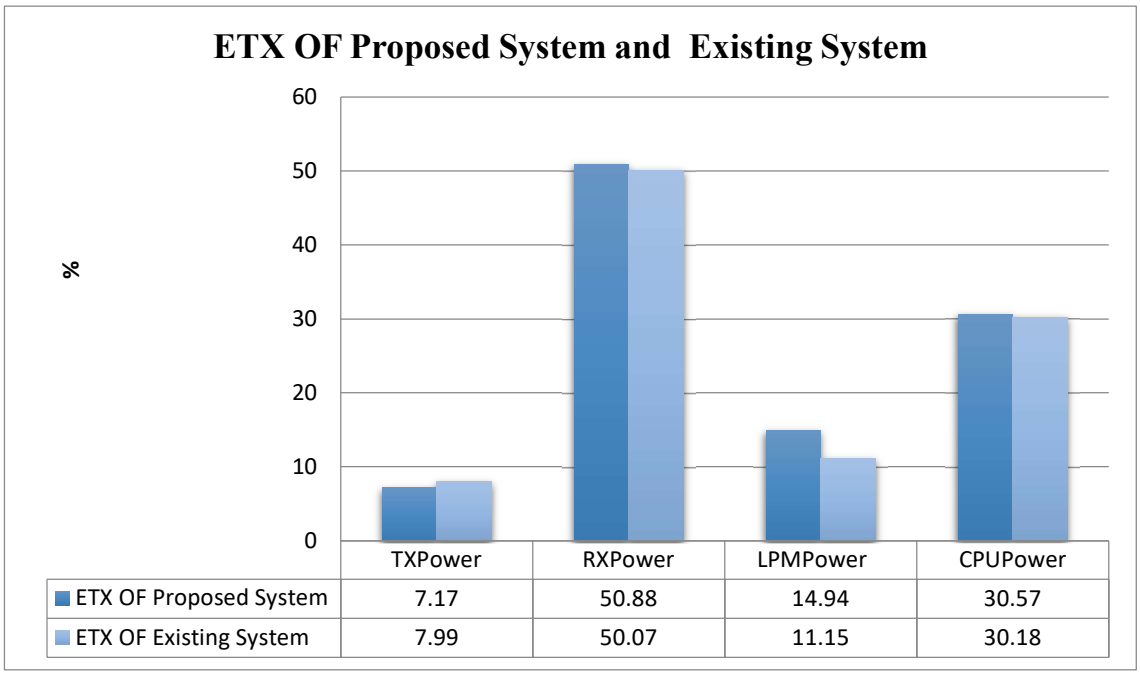

Figure.14 ETX OF Proposed System and Existing System

CONCLUSIONS

Discussion and a list of possible goals the results that were seen show that the proposed method has better performance than Hop Count and ETX in terms of power consumption and control traffic over head in fog computing network. From what we've seen in general, the objective function Hop Count works better in smaller networks, while ETX works better in bigger networks. ETX uses less power than Hop Count because it can have more reliable links than Hop Count. In this article, a RPL routing protocol With Modified Dijkhitra Algorithm is suggested for use in Internet of Things networks. It is suggested that a SP selection process be used so that the work can be spread out evenly among the parents on the list of parents. the DODAG creation process better by not sending DIO packets to nodes with lower rankings. This will lead to less energy use, which will make the network last longer. The suggested protocol offers a way to route data in a way that uses less power. This helps low-power nodes in local area networks (LLNs) keep their batteries charged. Using the MATLAB network simulator, several Results carried out to see how well the RPL routing protocol worked compared with existing technique protocols. The results show that our suggested protocol can save energy more effectively than other protocols, reduce the number of control packets, and make the IoT network last longer than with other protocols. One of our goals for the near future is to combine several metrics into a new objective function that can be used to choose and keep routing paths. One way to make the suggested protocol better is to think about how reliable the routing is. One of our goals for the future is to do real tests so that we can see how well the improved RPL technique works.

REFERENCES

1. Khandaker Foysal Haque, Ahmed Abdelgawad, Venkata P. Yanambaka, Kumar Yelamarthi College of Science and Engineering, Central Michigan University, Mount Pleasant, An Energy-Efficient and Reliable RPL for IoT MI, USA Email: haque1k@cmich.edu 978-1-7281-5503-6/20/$31.00 ©2020 IEEE
2. Ma, G.; Li, X.; Pei, Q.; Li, Z. A Security Routing Protocol for Internet of Things Based on RPL. In Proceedings of the 2017 International Conference on Networking and Network Applications (NaNA), Kathmandu, Nepal, 16–19 October 2017; pp. 209–213.
3. Silva, B.N.; Khan, M.; Han, K. Internet of Things: A Comprehensive Review of Enabling Technologies, Architecture, and Challenges. IETE Tech. Rev. 2018, 35, 205–220.
4. Humayun, M.; Niazi, M.; Zaman, N.; Alshayeb, M.; Mahmood, S. Cyber Security Threats and Vulnerabilities: A Systematic Mapping Study. Arab. J. Sci. Eng. 2020.
5. Airehrour, D.; Gutierrez, J.; Ray, S.K. Secure routing for internet of things: A survey. J. Netw. Comput. Appl. 2016, 66, 198–213.
6. Kharrufa, H.; Al-Kashoash, H.A.A.; Kemp, A.H. RPL-Based Routing Protocols in IoT Applications: A Review. IEEE Sens. J. 2019, 19, 5952–5967.
7. Le, A.; Loo, J.; Lasebae, A.; Vinel, A.; Chen, Y.; Chai, M. The Impact of Rank Attack on Network Topology of Routing Protocol for Low-Power and Lossy Networks. IEEE Sens. J. 2013, 13, 3685–3692.

8. Verma, A.; Ranga, V. Security of RPL based 6LoWPAN Networks in the Internet of Things: A Review. IEEE Sens. J. 2020, 20, 5666–5690.
9. Almusaylim, Z.A.; Alhumam, A.; Zaman, N. Proposing a Secure RPL based Internet of Things Routing Protocol: A Review. Ad Hoc Netw. 2020, 101, 102096.
10. Shafique, U.; Khan, A.; Rehman, A.; Bashir, F.; Alam, M. Detection of rank attack in routing protocol for Low Power and Lossy Networks. Ann. Telecommun. 2018, 73, 429–438.
11. Aris, A.; Örs Yalçın, S.B.; Oktug, S. New lightweight mitigation techniques for RPL version number attacks. Ad Hoc Netw. 2019, 85, 81–91.
12. Karthik, V.K.; Pushpalatha, M. Addressing Attacks and Security Mechanism in the RPL based IOT. Int. J. Comput. Sci. Eng. Commun. 2017, 5, 1715–1721. 16.
13. Mangelkar, S.; Dhage, S.N.; Nimkar, A.V. A comparative study on RPL attacks and security solutions. In Proceedings of the 2017 International Conference on Intelligent Computing and Control (I2C2), Coimbatore, India, 23–24 June 2017; pp. 1–6.
14. Raoof, A.; Matrawy, A.; Lung, C.-H. Routing Attacks and Mitigation Methods for RPL-Based Internet of Things. IEEE Commun. Surv. Tutor. 2019, 21, 1582–1606.
15. Glissa, G.; Rachedi, A.; Meddeb, A. A Secure Routing Protocol Based on RPL for Internet of Things. In Proceedings of the 2016 IEEE Global Communications Conference (GLOBECOM), Washington, DC, USA, 4–8 December 2016; pp. 1–7.
16. Airehrour, D.; Gutiérrez, J.A.; Ray, S.K. SecTrust-RPL: A secure trust-aware RPL routing protocol for Internet of Things. Future Gener. Comput. Syst. 2019, 93, 860–876.
17. Le, A.; Loo, J.; Luo, Y.; Lasebae, A. Specification-based IDS for securing RPL from topology attacks. In Proceedings of the 2011 IFIP Wireless Days (WD), Niagara Falls, ON, Canada, 10–12 October 2011; pp. 1–3.
18. Kamgueu, P.O.; Nataf, E.; Ndie, T.D. Survey on RPL enhancements: A focus on topology, security and mobility. Comput. Commun. 2018, 120, 10–21.
19. Dian Rachmawati1* and Lysander Gustin2* Analysis of Dijkstra's Algorithm and A* Algorithm in Shortest Path Problem Journal of Physics: Conference Series 1566 (2020) 012061 IOP Publishing doi:10.1088/1742-6596/1566/1/012061
20. Aris, A.; Oktug, S.F.; Yalcin, S.B.O. RPL version number attacks: In-depth study. In Proceedings of the NOMS 2016—2016 IEEE/IFIP Network Operations and Management Symposium, Istanbul, Turkey, 25–29 April 2016; pp. 776–779.
21. Dvir, A.; Holczer, T.; Buttyan, L. VeRA—Version Number and Rank Authentication in RPL. In Proceedings of the 2011 IEEE Eighth International Conference on Mobile Ad-Hoc and Sensor Systems, Valencia, Spain, 17–21 October 2011; pp. 709–714.