ISSN 2347 - 3983
Volume 8. No. 2, February 2020
International Journal of Emerging Trends in Engineering Research
Available Online at http://www.warse.org/IJETER/static/pdf/file/ijeter37822020.pdf
https://doi.org/10.30534/ijeter/2020/37822020

Enhancing Fault Tolerance of IoT Networks within Device Layer

Dr. JKR Sastry[1], Mr. Bhupathi[2]
[1]Koneru Lakshmaiah Education Foundation (Deemed to be University), Vaddeswaram, India, drsastry@kluniversity.in
[2]Koneru Lakshmaiah Education Foundation (Deemed to be University), Vaddeswaram, India, bhupathi@kluniversity.in

ABSTRACT

IoT based systems are error-prone and fragile, leading to the creation of faults in the entire network, causing misbehaviour many times. Many types of faults do happen within IoT networks due to the node, link, protocol conversion, and communication failures. Failures can happen due to malfunctioning of hardware and software installed into the devices. Among all network failures due to failure of nodes or links are more serious. A network topology which considers alternate links and redundant devices greatly improves the reliability of an IoT network.

An IoT network as such, built through different inter-linked layers which include Device, controller, restful services, gateway, internet and storage, and computing layers. Most of the studies in the literature have considered single topology generally hierarchical for connecting devices situated in different layers. The fault tolerance level of an IoT network dependent on the topology used as many issues such as alternate paths for communication, use of redundancy, and many such factors considered while building a network. A single computational model normally used for computing the fault tolerance level of an IoT network. Sometimes the fault tolerance computing model chosen may not suit the topology used for building a network for a specific layer within an IoT network. Use of different topologies suiting a sub-network in a layer and choice of different fault-tolerance computing models will help accurately determine the fault tolerance level of an IoT network.

In this paper, a Fault computing model that considers different topologies for developing sub-nets in different layers and computing models suited to a specific topology presented. The improvement in the fault tolerance of an IoT network achieved through consideration of two topologies, which include hierarchical and butterfly networks presented in this paper.

Key words: IoT network, Network Topologies, Fault tree analysis, Statistical models for computing reliability, butterfly topology, hierarchal topological models.

1. INTRODUCTION

IoT technologies are the next generation of technologies after internet technologies. IoT technologies are dynamic and differ from conventional networks. IoT networks must be scalable, maintainable, and Fault-tolerant. Many applications are being built these days using the IoT based networks, including the home automation, defense, and surveillance systems.

Every device in an IoT network fails and therefore needs to be made fail-free. Failure as such can happen due to breakdown, malfunctioning, or security leakage. Failures in IoT networks can happen at the device level, local network level, controller level, gateway level, internet level, and remote storage and computing level. Wherever the failure happens, the IoT network shall become in-operational and serves no purpose. IoT systems must be fault-tolerant, or else entire investment will go to waste and sometimes leads to disasters.

Fault tolerance as such could be as an integral part of the design of IoT based system. Fault tolerance must be in-built as part and parcel of the very IoT system itself. A typical IoT system must cater for implementation of many of the fault-tolerant strategies that have, in the literature.

In this paper, a hybrid topology presented that considers a butterfly topology at the device level, and a hierarchical topology within other layers with both the topologies interconnected forming a composite topology.

The fault tolerance of IoT networks greatly improves when networking is done using butterfly topology. IoT is a network of physical objects or ‘things’ that can interact with each other to share information and take action. The Internet of Things (IoT) is the interconnection of uniquely identifiable embedded computing devices within the existing Internet infrastructure.

Every device in an IoT network fail and therefore needs to be made fail free. Failure as such can happen due to breakdown, malfunctioning, or security leakage. Failures in IoT can happen at any level of an IoT network. Wherever the failure, the IoT shall become in-operational and serves no purpose.

Fault tolerance as such realized as an integral part of the design of IoT based system. Fault tolerance must be in-built as part and parcel of the very IoT system itself. It is necessary to find and identify various strategies that one can adopt to ensure a very high level of fault tolerance of the IoT based system.

Faults within any network are bound to happen due to various reasons. A network called fault-tolerant when it functions even normally when faults occur while the network is in use. IoT networks are fragile and therefore, must be made fault-tolerant. IoT networks used in the medical domain must be fault-tolerant as any misinformation flow will cause a devastating effect even to the extent of loss of human life. A small fault may lead to serious negative results.

When a Fault happens, generally, the data acquired is lost. Data must be preserved and retained at any cost. Use of Non-volatile memory within IoT based systems will help in recovering from the loss of the normal operation when a fault occurs. Fault tolerance is essential even at the cost of incurring overhead die to use of non-volatile memories.

The common approach to enhance the fault tolerance is Making a process to be running through several instances and adding many devices in parallel such that when one fails, there is another instance/device to take over. Computing fault tolerance is as such complex due to the existence of many intricate issues.

Fault tolerance of network generally expressed quantitatively in terms of success or failure rate is the rate of failure of topmost nodes existing in a Fault Tree — the success rate computed as 1 – Failure Rate. In a typical network success rate is the probability that at least one transmission path exists from a transmitting device to the destination device — the failure rate obtained by subtracting the success rate from 1.

An IoT network typically contains many layers such as device, controller, restful service, gateway, internet and storage & and computing layers. Many devices are interconnected through the realization of a subnet in each of the layers generally using the same topology. The topology to be used as such is dependent on the kind of devices used and the fault-tolerant characteristics of those devices. A single topology as such may not be suitable for all layers of the IoT network. The network can be designed using different network topologies and architecture and different implementation methods and a certain level of redundancy built into the system.

Many types of faults happen within IoT networks, and each of the faults must be considered and find the methods to mitigate the same. IoT networks typically are recognized into several layers. A fault-tolerance computing model used could differ from layer to layer. Fault tolerance of a network generally computed using a single computational model. A single computational model is generally not stuffiest as the networks in each layer may have deterministic or probabilistic behavior.

Choice of a topology suiting to the fault tolerance level of the devices contained in each layer and choice of the proper method to compute fault tolerance of a sub-net will lead to high fault-tolerant IoT network. In this paper, a composite model that considers different networking topologies and fault tolerance computing models that enhance the fault tolerance level of an IoT network presented.

2. RELATED WORK

Maheswari et al., [1], have presented different kinds of failures that can happen in a mobile network that include power failures, energy failures, and network failures such as node and link failures. They have presented different techniques considering a subset of a set of failures and have shown the reliability of the network and the way the reliability enhanced through consideration of other aspects of fault tolerance that include alternative power, energy, and the network management.

Generally, mission-critical real-time systems implemented through distributed embedded systems. The real-time characteristics of an embedded system mapped to the requirements of a distributed system which are dynamic. Most of the techniques available for computing the fault tolerance of a system don not considered the distributed considerations of a system. FTA based systems consider every working component and the connectivity between them, whereas the distributed systems built through logical models that describe connectivity between the components.

Paul Rubel et al.,[2], have presented approaches /techniques using which FTA applied for computing the fault tolerance of distributed embedded systems. They have considered three FT based techniques/ approaches that include auto-configuration of dynamic systems, mixed-mode communication, and maintenance of redundancy into peer-peer communication. They have described an integrated system that combines an off the shelf middleware with different FT based techniques that have been the advanced models implemented by them.

Choreography is a mechanism generally used to define object interaction dynamically not withholding any of the statically defined object linkages. This technique generally affects the coherence that exists between the objects. Due to this reason, there could be loss of messages flowing across various objects contained in an application. There could be several faults occurring due to this reason leading to the failure of a system. Sylvain Cherrier et al.,[3], have proposed the method that synchronizes, de-synchronizes and a re-synchronizes the objects such that coherence between the objects intact leading to failure-free systems while dynamically configurable systems implemented through the process of choreography.

All the devices in an IoT network interconnected as a subnet in the bottom-most layer of the network. The protocols used for effecting communication between the devices are also pre-identified and taken in to count while designing the IoT based systems. In this process, there could be a possibility that unlike devices may be connected leading to the generation of unwanted faults during the working of these devices. On the other hand, Chen Wang1 et al.,[4], have recommended the analysis of data generated by the respective devices and established/predict the logical relationships between those devices which can be used as a basis to predict faults and maintenance requirements of an application/objects. Generally, this needs fault diagnosis and in a way, enhancing the fault tolerance/reliability through periodic maintenance of the devices which are predicted to be error-prone.

WSN networks are fault-prone due to loss of communication link, loss of data during transmission and, missing sensor nodes, etc., due to the occurrence of various factors such as asymmetric communication links, dislocation of sensor node and collision, radio interference, environmental impact, and power depletion. There are several mechanisms presented in the literature that includes cluttering, inducing redundancy, deployment of objects dynamically to mitigate the failures that can happen within WSN networks. Gholamreza Kakamanshadi et al., [5], have presented an analysis of the techniques considering the weakness and strengths of the mechanisms and arrived at suitable mechanisms deployed given a composer of failure situations.

Cloud computing technologies deal with a large amount of data, so it is cost-effective for implementing IT-based solutions. Many issues are to be addressed considering the usage of the cloud. Among all, fault tolerance and securing the data are the most important issues. DBK Kamesh et al., [6], have presented that a fault occurring in one device might lead to faults occurring in one or more connected devices. They have implemented a design method to achieve high reliability, which leads to improvising the fault tolerance of the networks that connect clouds.

Customers are using Cloud computing for meeting their IT requirements. However, the users are concerned with the security and availability of the data as cloud computing infrastructure can be affected due to attacks by malicious users and due to the generation of different types of faults that happen due to failure of either Hardware or software. Susmitha et al., [7], have discussed the challenges that one should address while using cloud computing for meeting their IT requirements. One such challenge is to create fault tolerance within a network that connects various physical and logical resources. An architectural framework has been recommended implementation of which will provide fault tolerance within the network

For developing an IoT network, three things focused; the network should be efficient, economical, and robust. Kai Fan et al.,[8], have presented random topologies, which promises high performance by reducing the cost of network establishment. It automatically explores to build temporary routing when unpredicted failure occurs, which will not affect the overall network. By implementing these methods, they have improved the fault tolerance and availability of the Networks.

Huge data is collected using the IoT network, which is made available to several local and remote users. Routing the information across the IoT network must cater for faults that may occur while the IoT system is in running state. Zaki Hasan et al., [9], presented a routing algorithm which is capable of constructing and recovering and also selecting k-disjoint paths that are fault free and then communicate the data across those few selected fault-free paths, The authors considered optimization of energy required to communicate the data across the network while ensuring that minimum delay in communicating the data. They have compared PMSO with other similar algorithms and shown the efficiency and effectiveness of the algorithm.

The architecture of an IoT network designed considering the possibility of occurrence of the faults within the network. A fault-tolerant architecture proposed by Asad Javeda et al., [10], used for implementing a variety of IoT based applications. In the architecture, they have considered the placement of software stacks at different locations for making deployment decisions at run time. They have also considered many other issues such as long-distance network connectivity, faults happening within edge devices, harsh operating environment, etc. In the architecture that included the issue of processing that should take place at both the edges of devices and the cloud.

Implementing a fault-tolerant IoT based system is complex as one has to deal with many of the dynamically evolvable and coupled systems. Alexander Power et al., [11], built a framework using Micro-services. In the framework, they have included the support required for the IoT system to tolerate the faults when they happen through the inclusion of machine learning processes. The machine learns when the faults happen and then take tolerant actions immediately so that the network will fail free.

A cluster or a leader node used for communicating within the IoT, WSN, and Adhoc networks. The node must be selected such that it has maximum energy or located to the extreme left of the network such that it would be the last node. If the head node or the leader node fails, the entire IoT network will fail. Routing algorithms are the key to any communication. Routing algorithms must be intelligent to elect a cluster head when a fault happens such that fail free communication happens. Achene Bounceur1 et al., [12], have expressed that the leader must be elected dynamically

considering the paths that must have failed. They have presented an algorithm for electing a leader through the use of a local minimum as a root and the concept of flooding is used to determine a spanning tree for routing the communication over the spanning tree. The two spanning trees coincide, the better one is selected, and the other ignored. The root of the spanning tree will be the leader through which the communication is affected.

A cloud-based IoT network architecture proposed by Jatinder Grover et al., [13],. The architecture built with the components required for making the network survive even in the presence of failure of the edge servers. The network recognized as different hierarchies, and the communication is re-directed to different hierarchy when a fault noticed in a different hierarchy. They have included mobile agents on the servers that share the system states, data, and other agents if the system fails at fog, edge, mist, or cloud. Inclusion of these components will help re-direction in the case of any server failure.

IoT is a layered network which is having different layers; it deals with many heterogeneous subnetworks. Failure rates of an IoT system are dependent on network topology as the faults can happen within the network hardware device and even can happen in the software that runs in different layers. Every IoT based must be scalable, maintainable, and highly reliable. Failure of an IoT system will lose its identity and leads to customer dissatisfaction. One has to implement quite number strategies to make an IoT system more reliable. Many authors considered the reducing levels of the performance as a kind occurrence of faults with IT and therefore performance of an IoT system must also be considered for assessing the fault tolerance of the IoT based system[14][15[16][17][18][19][20][21][22][23][24][25] [26] [27][28][29][30][31].

3. COMPARATIVE ANALYSIS

Table 1 shows the comparison of findings by various researchers relating to fault tolerance of IoT networks that are established either through wired, wireless, and those networks connected through backend cloud computing infrastructure. Many mechanisms have been proposed in the literature using which fault tolerance can be improved. The Mechanisms include redundancy, clustering, and deployment based mechanisms. The after-effects of improving the fault tolerance in terms of higher availability increase in accuracy, savings in energy, enhanced network life, minimization of the failures of the components, increase in efficiency, and robustness explained in the literature.

The issues of fault tolerance have been addressed considering the kind of faults that can happen and mechanisms to mitigate the same without much consideration to the extent to which fault tolerance can be improved. The focus of the researchers, however, is on the failures of nodes, links, routers, and communication, which all get culminated into topologies which take into account all aspects of failures of the networks.

Building fault tolerance at the topology level takes into account almost all aspects related to fault tolerance. In the literature, the dynamic adaption of network topologies suggested which is very complicated for implementation as the IoT network as such is heterogeneous, especially within the bottom level of the network

In this report, a methodology has been presented taking into account the static nature of the IoT networks considering improving the fault tolerance in terms of establishing alternative networking, especially looking at the device level of the IoT networks.

4. PROTOTYPE IoT NETWORK FOR EXPERIMENTATION AND RESULTS

4.1 Overview of prototype IoT network

An IoT network typically contains several layers of networking that include Device Layer, Controlling Layer, Services layer, gateway Layer, and cloud computing Layer. The IoT network must be fault-tolerant at every layer. In this paper, an approach has been built considering all the layers in the network while exploring the fault tolerance in device layer while assuming that the fault tolerance of the layers in the network fixed and no variances noticed in those layers.

A typical IoT network developed for carrying the experimentation shown in **Figure 1**. The IoT network has been built considering all the layers situated in a typical and comprehensive IoT network. The network is built considering all the layers that include device layer, controller layer, services layer, gateway layer, and computing layer

Device Layer:

One has to face many challenges to implement a fully operational IoT based system all the time. One has to look into several issues which include connectivity between the devices, power management, scalability, interoperability, security, and availability. Management of the IoT devices by using standard protocols by using the standard services rendered by third-party vendors is the key issue.

Efficient management of the device will lead to proper integration, monitoring, organizing, and remotely controlling the functioning of the devices through the provision of internet-enabled or interfaced devices. These kinds of implementations will help to implement the required redundancy, fault tolerance, security, and connectivity of the IoT devices and as such, helpful in managing the entire life cycle of the devices. Some of the important processes included in an IoT based system include

authentication/authorization, registration, provisioning, monitoring and diagnostics, configuration, troubleshooting, etc.

Unlike other system, IoT systems are application dependent. Based on the kind of application implemented, one has to select appropriate communication, networking, and connectivity protocols used with the devices that are internet enabled. The users have many options at their hand for selecting different connectivity and communication protocols, which include MQTT (Message Queuing Telemetry Transport), DTLS (Datagram Transport Layer Security), CoAP (Constrained Application Protocol), etc. Many options also exist for selecting wireless communication protocols that include Bluetooth, Wifi, Zigbee, LPWAN, IPv6, RFID, NFC, Z-Wave, etc., Communication between the devices can also be affected through the use of Satellite, cellular and Ethernet-based optic fiber communication. Each of the options has a bearing on bandwidth, range power requirements. The devices that participate in an IoT based communications must be selected considering all those aspects above.

Controller Layer:

The middle tier of an IoT Based system is the controller layer. The components included in the middle tier are primarily responsible for data collection; aggregations and transmission of the data form the devices situated in the lower tier of the IoT based system. The controller is primarily responsible for collecting the data, processing, and then transmission of the same to back end systems where the information is stored. The periodicity of data collection, processing, and transmission depends on the streaming of the data in real-time, and the speed with which buffering, encoding, decoding done. A certain amount of processing is also done on the collected data based on pre-defined rules, regulations, and policies and actuating if any required will be locally triggered. Many types of controller exits which vary a lot based on the availability of interfaces to get connected with the devices, availability of security provisions and the kind of devices that can be connected to deal with various environmental factors. The devices can be connected to the controllers through different networking options that include LAN, USB, WAN, CAN, RS485, I^2C, and firewire.

Services Layer:

The data collected by the controller can be used for processing locally or moved to other devices incoherent manner. It has been a serious challenge to collect a large amount of data at one location and move the same to remote locations, where It gets stored and processed. The controller is responsible for sending the data to remote locations. Since data communication power of a controller is a week, the data routed through a server called services server. Remote users can route their service request to this server through the internet, Cloud-based users also can get the access the data stored in the cloud transmitted through services server.

Gateway Layer:

There can be multiple paths to the cloud. The service's servers have the option of sending the data through any chosen path to the cloud. While the server supported with few communication interfaces, the cloud might have several interfaces, in which case a gateway implemented. The main aim of an IoT network is to connect thousands of devices that collect and send data through the internet. Several of the communications systems implemented that include Cellular (2G-5G), wireless (Wi-Fi, ZigBee), wired (Ethernet), etc. A local network connected to a gateway through wireless and wired communication systems.

Cloud / Application Layer:

The topmost layer in an IoT based network is the applications layer in which several of the user applications situated. Users interact with a chosen application. The user-defined applications situated on mobile phones, laptops, or desktops. In this layer, several tools used for different purposes such as connecting, communicating, displaying, and visualizing any intelligent applications such as logistics management, intelligent transportation, disaster recovery, ensuring safety, etc. the 5 Layer architecture used for building IoT based networks provides a framework used for developing different types of networks. With the implementation of IoT based networks internet is being moved away from transmitting data to providing different kinds of services. WEB servers predominantly used for providing the requested services to the end-users, The services layer generally built with a WEB server such that user requests serviced by the WEB server

4.2 Hardware Specification – Prototype IoT network

The prototype IoT network has been developed using 3 Arduino-Uno Microcontrollers that are used to collect three types of data that include Temperature, Light using operation of three devices that drive, FAN, Light and Air conditioner. Two sensors used for detecting the temperatures and Light (DHT11, LDR). All three Arduino – Uno microcontrollers are interfaced with Wi-Fi communication modules externally.

All these controllers are networked to form into clusters through two cluster heads established through “Node MCU” Microcontroller that has inbuilt Wi-Fi interface to establish communication in-between the cluster head and the data acquisition systems and the node MCU has been built using externally interfaced ZigBee systems in addition to its internally situated Wi-Fi system. The dual communication systems using Wi-Fi and ZigBee provides for reliable communication between the Cluster heads and the controller.

The controller function implemented through Raspberry Pi Microcontroller that has in-built Wi-Fi and ZigBee communication. The cluster heads communicate with Microcontroller through dual communication channels that include ZigBee and Wi-Fi communication protocols. The controller connected to the restful server through two communication channels, which include Ethernet and Wi-Fi communication systems in peer-to-peer communication mode. The restful server is built using Intel i7 technologies which run windows operating system. A restful services server implemented on this machine. Three services developed include FAN service, AC service, and LIGHT service.

These services triggered from outside using a service request initiated through a SOAP protocol which is an XML code. The XML code is interpreted to find the service requested by the external user. The SOAP request is parsed to find the service which translated to a command which is then transmitted to the controller to execute the command and the result of the execution of the command transmitted to Restful services server which intern communicates the service outcome to the external user through the gateway.

An application running on the internet can make a request to Restful service server for want of service. The user application can also initiate a request for data through cloud computing infrastructure through the invocation of SaaS service or initiating a request to the WEB server for want of data. The interface between the User application and the restful service server achieved through either a gateway or using the direct connection using a Wi- fi Channel for imitating a service request The gateway developed through three different communication modes both the input and output interface which include ZigBee, Wi-Fi, and GSM and Ethernet. The gateway is intelligent to receive input from one of the active input channels and translate the protocol to one of the active, outgoing interfaces, thus working as an active communication protocol converter. The gateway is designed to consider the communication speed, buffering, arbitration, timing, etc. The outgoing channels of the Gateway connect to the application, WEB server, and the cloud computing infrastructure.

The restful service server invokes a WEB service or a cloud computing service to submit the data collected by the device cluster especially through the invocation of web server service or a cloud computing service for storing the data in a database. The application is also built to retrieve the data from the database for doing analytics if any, by using one of the standard statistical tools.

5. INVESTIGATIONS AND FINDINGS

5.1 Fault Analysis of IoT networks

Given an IoT network, analysis has to be done to find the fault tolerance strength of the network. The faultiest used technique is Fault tree Analysis. Fault tree analysis is an analytical technique. In this approach, an undersized state of the system is defined and then the same is analysed in terms of environment, operation, safety, criticality, etc. and then find different ways in which the undesired event can occur. A fault tree is a graphical model that has all combinations of the faults, both sequential and parallel that can occur, leading to an undesirable event. The faults as such can be hardware faults, network faults, software faults, or faults occurring due to human error. The basic interrelations between the faults and the events are depicted using a fault tree. The undesired event will be the top node of the fault tree. A fault tree is not a model that can capture all system failures or that causes that lead to system failures.

The top node of a fault tree relates to the occurrence of a specific event, which is a kind of system failure. The faults tree deals with those faults that lead to the top event. There can be many and many faults that could be related to the top of the event, making the construction of the tree complex. To avoid this few venerable and most important faults are selected and modelled into the tree. AND gates and OR gates are used to show the relationships among the faults that can occur on different devices. A fault tree model is not a quantitative model, and In fact, it is a qualitative model that can be measured quantitatively

In the fault, tree gates used for passing through the effect of the faults up the ladder to reach the root node. The relationship between the events modelled through the gates. It shows how the lower order events trigger higher-order events. The out from a gate is the higher-order event. The lower order events are the inputs to the gates. The gates are not like logic gates. The gates are just symbolic to show what output event raised due to the occurrence of the lower order events. The occurrence of an output event due to the occurrence of one or more input events modelled through the OR gate, the occurrence of the output events when all input events occur modelled through AND gate.

Assessing fault tolerance of IoT networks is required as the devices in the network are fragile and lightweight. The fault tolerance of an IoT network is majorly dependent on the way various hardware elements are interconnected and the kind of devices selected for achieving the network. Network topology is the most important aspects considered from the perspective of fault tolerance of the IoT network. The topology as such takes care of many failure conditions that can generally happen within an IoT network.

Many methods/models are in existence for computing the reliability of any given network, the most important being reliability analysis through fault tree analysis and probability models.

5.2 Estimating Fault Tolerance of an IoT Network using the FT Analysis

FT analysis carried on the prototype model, and the derived FT diagram for the prototype model, a part of which shown in **Figure 2**. The relationships among different elements that form the network are connected through OR and gates to simulate the failure model of the prototype IoT network. Fault computations carried through compilation of failure rates of the devices and the failure of one device due to failure of other devices based on the relationships that exist among the devices through AND or OR relationships among the devices. The fault computations are undertaken using a bottom-up approach until the root node arrives. The failure rate of the root node is considered to be the failure rate of the IoT network. The failure rates of each of the device obtained through Manufacturer data. Table 2 shows the fault commutations of the Prototype model. From the table, one can observe that the success rate of the prototype IoT model is 0.729. The failure rate of the prototype model will then be 1 – 0.729 = 0.271.

5.3 Modifying the IoT network with standard networking topology

The prototype model discussed in section 3 is a pure hierarchical networking diagram and as such, do not follow any specific topology. Many networking topologies exist in the literature that includes Butterfly, crossbar, multi-stage, mesh, and such other combination of topologies. It is easy to compute the reliability of these networks due to the existence of probability-based computational methods for computing reliability.

The failure rate of the network reduces by implementing different topologies than the tree topology used for the sample IoT network. In a multistage network, there will be multiple stages at which the inputs are processed to transform inputs to outputs. The switching nodes sutured in different processing stages to not all undertake to process, but all move the processed outputs to the nodes situated in a different stage.

The network is built using N X N switches. Each switch takes N inputs and produces N Outputs. The switches interfaced through different connections that include the cross, straight, lower broadcast, and upper broadcast. A 2 X 2 network is called the butterfly network.

The devices are networked using Butterfly model. The Links that are at a distance two are connected to switch box. A 4 X 4 butterfly network achieved through two 2 X 2 Butterfly networks. Figure 3 shows a 4 X 4 Butterfly network.

Consider Φ(0), which is the probability that at least one line out of a switch box at the output stage is functional then the probability is 1-$q_{l,}$ where q_l is the failure rate of the link.

$$\phi(i) = 1 - ql \qquad (1)$$

Where k = Number of stages, ρ_l = probability that a link fails and Φ (i) is the probability that a switch box in stage K can fail. Φ(k) can be computed using equation (2).

$$\phi(i) = 1 - (1 - pl\phi(i-1))2 \qquad (2)$$

Due to the simplicity of the butterfly topology, it has been contemplated to modify the original prototype IoT network using Butterfly topology to assess the change in the fault tolerance state of the IoT network. Figure 3 shows the revised network topology.

5.4 Modifying the Prototype IoT network into a Hybrid Topology model

The Prototype IoT network is modified using two topologies. While the butterfly model used for building a network within the Device level of the IoT network, the other layers of the network are connected using hierarchical networking topology. The reliability computation of such a network achieved through probability model prescribed for Butterfly networks and by conducting FTA on the Hierarchical model. The composite Fault tolerance computational model presented that include both the computational models; Butterfly model at the device level and hierarchical models at other levels of the IoT network. The revised IoT network placed in **Figure 4**.

5.5 Computation of Fault tolerance at device Level using the butterfly network

An IoT network developed in terms of different layers of the networks which get connected hierarchically. The layers generally include Device, Controller, Services, Gateway, and cloud computing layers. In the work modification of networking, topology is considered at device level only while keeping the rest of the network static and hierarchically laid.

In the revised working diagram, a 4X4 butterfly network has been constructed at the device level considering 3 Inputs (AC, FAN, and LIGHT. The reliability of the device level network computed through the use of probability models discussed in section 4.2. Following are the reliability computation of the butterfly network built at the device level. Using equation (1) and equation (2)

$$pl = 0.98, ql = 0.02$$

$$\phi(i) = 1-(1- pl\phi(i-1))2$$
$$\phi(1) = 1-(1-0.98\phi(1-1))2$$
$$= 1-(1-0.98\phi(0))2$$
$$= 1-(1-0.98(1-ql))2$$
$$= 1-(1-0.98(1-0.02))2$$
$$= 1-(0.02(0.98))2$$
$$= 1-(0.0196)2$$
$$= 1-0.00038$$
$$= 0.998$$

The fault tolerance of the cluster head which receives the outputs out of the device level network thus computed to be of value 0.998. The Cluster heads considered as devices in a hierarchical network having a fault tolerance value 0.998.

5.6 Computation of Fault tolerance of the entire network

The modified network thus can be further modified, as shown in **Figure 4,** considering that the process flow commences from the cluster head. A fault tree analysis conducted on the modified IoT network. Figure 5 shows the faulty tree Analysis Diagram. The fault tolerance computations are shown in **Table 3,** considering the FTA diagram shown in **Figure 6**. From the computation can be seen, the fault tolerance value is improved from **0.79 to 0.90** due to the usage of the butterfly model at the device level.

5.7 Comparative Analysis of Fault computations of IoT Network

An analysis of fault tolerance computed through Fault Tree analysis of original network, fault tolerance computed through FTA Analysis and Probability model is placed below in **Table 4**. From the table, one can see that the reliability of an IOT based network increased tremendously when IoT network established as a Butter Fly network. Even the complexity of the network decreases tremendously. 21 Communication modules are reduced to 4 Communication models when networking carried as per Butterfly networking architecture. In the case of a hierarchical star topology, it is not possible to represent the operational aspects of the network as probability models. Even if an attempt made, it would be a too complicated model which makes the computations quite complicated.

Table 4: Comparative Analysis of Reliability assessment considering different topologies

Type of topology used	Success rate based on FTA Analysis	Success Rate Based probability models
Hierarchical star Topology	0.729	--
Butter Fly Topology	0.900	0.900

6. CONCLUSIONS

IoT networks are fragile that connect numerous devices using many protocols and interfaces. IoT networks are lightweight, and fault tolerance levels of such networks are quite low. The reliability of IoT networking is quite dependent on the kind of topology, and the way networking carried.

IoT networks contain many layers, and a sub-net exists in each layer connecting the devices used in a specific layer. Networking of the devices in the sub-nets generally done using the same topology and establish the connectivity between the sub-nets hierarchically. Fault tolerance of an IoT network depends on the topology used as the topology dictates the availability of alternate paths of execution, availability of the redundancy fault tolerated switching system, etc., One has to choose a suitable topology based on the kind of devices connected within the sub-net. Choice of a proper topology in each of the layer is the key that dictates the overall fault tolerance of the IoT network.

Choice of a Butterfly network at the device level and hierarchical topology at the higher levels improved the fault tolerance of the prototype model. The success rate of the IoT network enhanced by about 17% moving from 0.729 to 0.900. The success rate is consistent, considering the computations undertaken through FTA and Probability models.

Choice of a computational model for computing the success rate is also very important to arrive at realistic estimates. Choice of probability model when it comes to the butterfly network and FTA model when it comes to hierarchical networks has been proved to be quite effective.

Developing an IoT network using butterfly topology greatly improves the fault tolerance level while reducing the complexity of the very network, especially in reducing the communication interfaces.

REFERENCES

1 D.Maheshwari1, A. Dhanalakshmi, Fault Tolerance in Mobile ad hoc Network: A Survey. *International Journal of Advanced Research in Computer Science and Software Engineering*, Volume 3, Issue 3, pp: 191-195

2 Paul Rubel, Aniruddha Gokhale, Aaron Paulos, Matthew Gillen, Jaiganesh Balasubramanian, Priya Narasimhan, Joseph Loyall, Richard Schantz, (2007). Fault-tolerant approaches for distributed real-time and embedded systems. *(DARPA) under contract NBCHC030119 https://ieeexplore.ieee.org/document/4455043,* pp: 1-8

3 Sylvain Cherrier, Yacine M. Ghamri-Doudane, Stéphane Lohier, and Gilles Roussel, (2014). Fault-

recovery and Coherence in the Internet of Things Choreographies. *IEEE WF-IoT, HAL Id: hal-00957056*, pp:1-7

4 Gholamreza Kakamanshadi, Savita Gupta, Sukhwinder Singh, (2015). A Survey on Fault Tolerance Techniques in Wireless Sensor Networks. *IEEE, 978-1-4673-7910-6/15/$31.00*, pp: 168-173

5 Chen Wang1, Hoang Tam, (2015). An IoT Application for Fault Diagnosis and Prediction, *IEEE, 978-1-5090-0214-6/15, DOI 10.1109/DSDIS.2015.97*, pp: 726-731

6 DBK Kamesh, JKR Sastry, Ch. Devi Anusha, P. Padmini, G. Siva Anjaneyulu, (2016). Building Fault Tolerance within Clouds at Network Level. *International Journal of Electrical and Computer Engineering*, Vol. 6, No. 4, pp: 1560-1569.

7 S. L. Sushmitha, Dr. D. B. K. Kamesh, Dr. J. K. R. Sastry, V. V. N. Sri Ravali, Y. Sai Krishna Reddy, (2016). Building Fault Tolerance within clouds for Providing Uninterrupted Software as Service. *Journal of Theoretical and Applied Information Technology*, Vol.88. No.1, ISSN: 1992-8645, pp: 65-76

8 Kai Fan, Jiapeng Lu, Dazhen Sun, Yong Jin, Ruimin Shen, Bin Sheng, (2017). Failure Resilient Routing Via IoT Networks. *978-1-5386-3066-2/17, DOI 10.1109/iThings-GreenCom-CPSCom-SmartData.*

9 Mohammed Zaki Hasan and Fadi Al-Turjman, (2017). Optimizing Multipath Routing With Guaranteed Fault Tolerance in the Internet of Things. *IEEE Sensors Journal*, *Digital Object Identifier 10.1109/JSEN.2017.2739188*. VOL. 17, NO. 19, pp: 6463-6473

10 Asad Javed, Keijo Heljanko, Andrea Buda, and Kary Främling, (2018). A Fault-Tolerant IoT Architecture for Edge and Cloud. *IEEE, 978-1-4673-9944-9/18*, DOI:10.1109/wf-IoT.2018.8355149, pp: 813-818.

11 Alexander Power and Gerald Kotonya (2018). A Microservices Architecture for Reactive and Proactive Fault Tolerance in IoT Syste*ms. IEEE, 978-1-5386-4725-7/18/$31.00, DOI:10.1109/wowmom. 2018.8449789*, pp: 1-6

12 Ahc`ene Bounceur, Madani Bezoui, Massinissa Lounis, Reinhardt Euler, Ciprian Teodorov, (2018). A New Dominating Tree Routing Algorithm for Efficient Leader Election in IoT Networks. *IEEE 978-1-5386-4790-5/18*, DOI:10.1109/ccnc.2018.8319292, pp:1-2

13 Jitender Grover and Rama Murthy Garimella, (2018). Reliable and Fault-Tolerant IoT-Edge Architecture. *DOI: 10.1109/ICSENS.2018.8589624*, pp:1-4

14 Murty, A. S. R., Teja, K., Naveen, S. (2018). Lathe performance monitoring using IoT. *International Journal of Mechanical Engineering and Technology (IJMET)*, Volume 9, Issue 4, pp. 494–501

15 Rambabu, K., Venkatram, N. (2018). Traffic flow features as metrics (TFFM): Detection of application layer level DDOS attack scope of IoT traffic flow. *International Journal of Engineering and Technology (UAE)*, 7(2), pp. 203-208

16 K., Prabu, A.V., Sai Prathyusha, M., Varakumari, S., (2018). Performance monitoring of UPS battery using IoT. *International Journal of Engineering and Technology(UAE)*, 7 (2.7), pp: 352-355

17 Poonam Jain, S., Pooja, S., Sripath Roy, K., Abhilash, K, Arvind, B. V., (2018). Implementation of asymmetric processing on multi-core processors to implement IOT applications on GNU/Linux framework. *International Journal of Engineering and Technology (UAE)*, 7 (2.7), pp:710-713

18 Raja Sekhara Naidu, G., Venkat Ram, N, (2018). Urban climate monitoring system with IoT data analytics. *International Journal of Engineering and Technology (UAE)*, 7 (2.20), pp: 5-9

19 Poonam Gupta, Kopparti Veera Venkata Satyanarayan, Dilip Devchand Shah, (2018). Development and testing of message scheduling middleware algorithm with SOA for message traffic control in the IoT environment. *International Journal of Intelligent Engineering and Systems.* Vol.11, No. 5, DOI: 10.22266/ijies2018.1031.28, pp: 301-313

20 Poonam Gupta, Kopparti Veera Venkata Satyanarayan, Dilip Devchand Shah, (2018). IoT multitasking: Development of hybrid execution service-oriented architecture (HESOA) to reduce response time for IoT application. *Journal of Theoretical and Applied Information Technology*, 96(5), pp. 1398-1407,

21 Yasaswini, A., Daya Sagar, K. V, Shri Vishnu, K., Hari Nandan V, Prasadara Rao, P. V. R. D., (2018), Automation of an IoT hub using artificial intelligence techniques. *International Journal of Engineering and Technology(UAE), 27DOI: 10.14419/ijet.v7i2.7.10250*, Vol 7, No 2.7, pp. 25-30

22 Ramaiah, C. H., Parimala, V. S., Kumar, S. P., Reddy, G. B., Rahul, Y. (2018). Remote monitoring through the tab. *International Journal of Mechanical Engineering and Technology*, Volume 9, Issue 1, January 2018, pp. 490–498

23 Y. Shanmukha Sai, K. Kiran Kumar, (2018). Internet of things and their applications. *International Journal of Engineering and Technology(UAE)*, Vol 7, No 2.7, pp. 422-427

24 J. Rajasekhar , Dr. JKR Sastry, An Approach to hybridisation of embedded system networks, International Journal of Engineering & Technology 7 (2.7) (2018) 384-389

25 T. Pavithra1, J. K. R. Sastry, Strategies to handle heterogeneity prevalent within an IOT based network, International Journal of Engineering & Technology, 7 (2.7) (2018) 77-83

26 Bhupathi, Dr. JKR Sastry, A framework for effecting fault tolerance

27 within IoT network, Jour of Adv Research in Dynamical & Control Systems, Vol. 10, 02-Special Issue, 2018

28 K.V. Sowmya1, Dr. JKR Sastry, Performance evaluation of IOT systems – basic issues, International Journal of Engineering & Technology, 7 (2.7) (2018) 131-137

29 Dr. J, Sasi Bhanu, Dr. JKR Sastry, P. Venkata Sunil Kumar, B. Venkata Sai, K.V. Sowmya, Enhancing Performance of IoT Networks through High

Performance Computing, Volume 8, No.3, May - June 2019

30 M. Sai Rama Krishna, J K R Sastry , J Sasi Bhanu, Building Fault Tolerance Within Wireless Sensor Networks: A Butterfly Model, Research Journl of Applied Sciences 12 (2): 139-147, 2017

31 J. K. R. Sastry, K. Sai Abhigna, R. Samuel, D. B. K. Kamesh, Architectural models for fault tolerance within clouds at infrastructure level, ARPN Journal of Engineering and Applied Sciences, VOL. 12, NO. 11, JUNE 2017

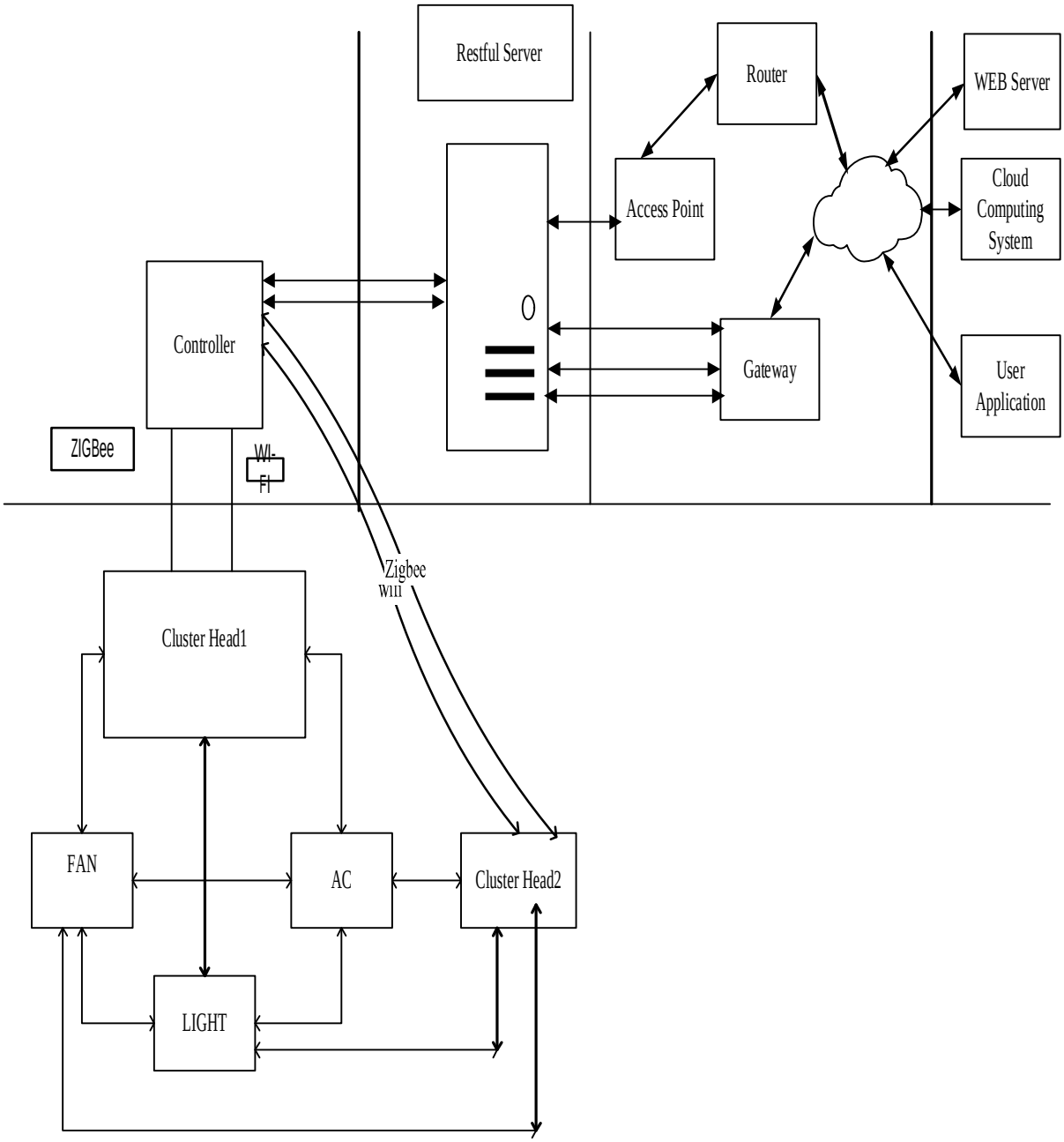
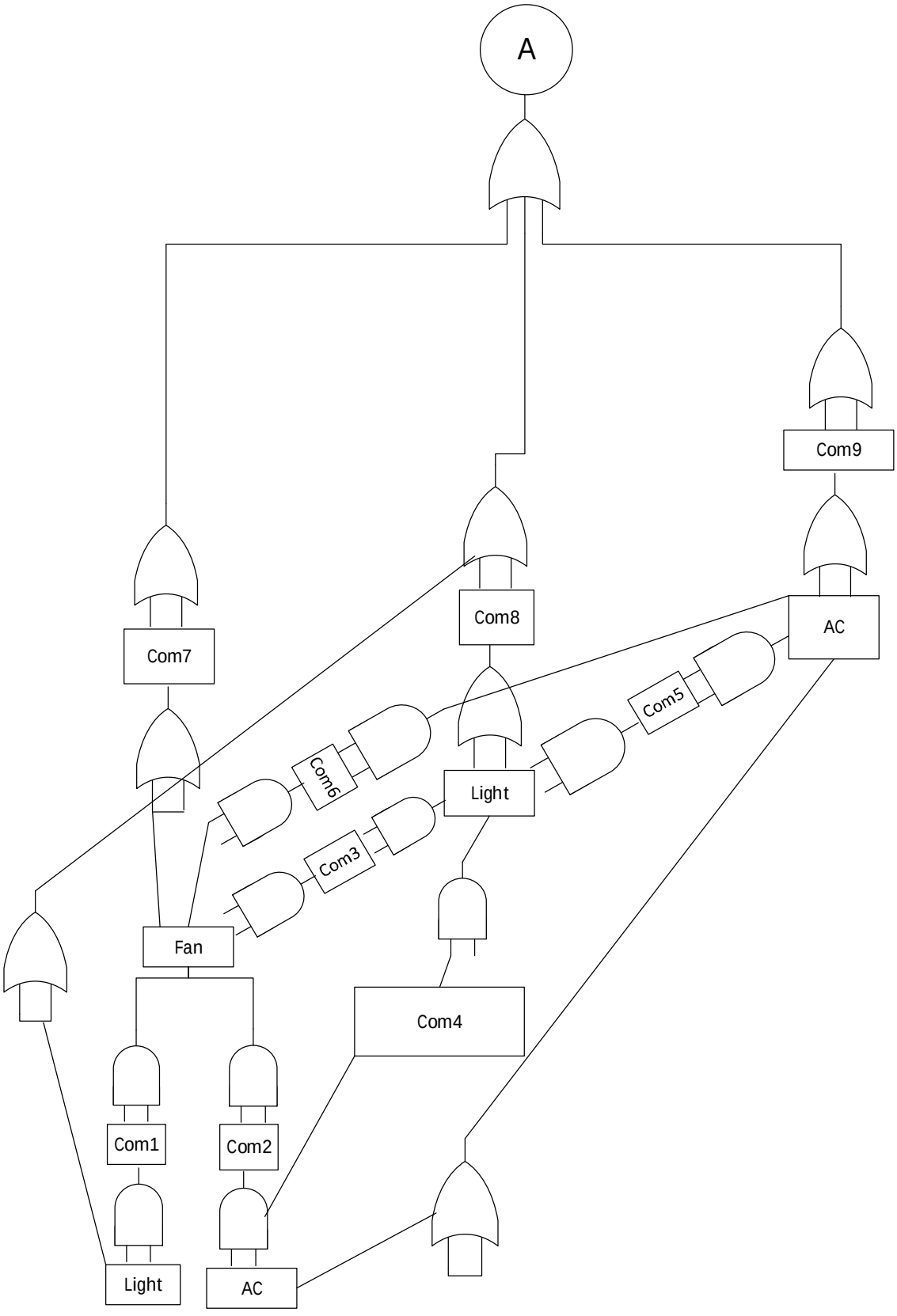


Figure 1: Prototype - IoT Network


A
Com9
Com8
AC
Com7
Com5
Com6
Light
Com3
Fan
Com4
Com1
Com2
Light
AC

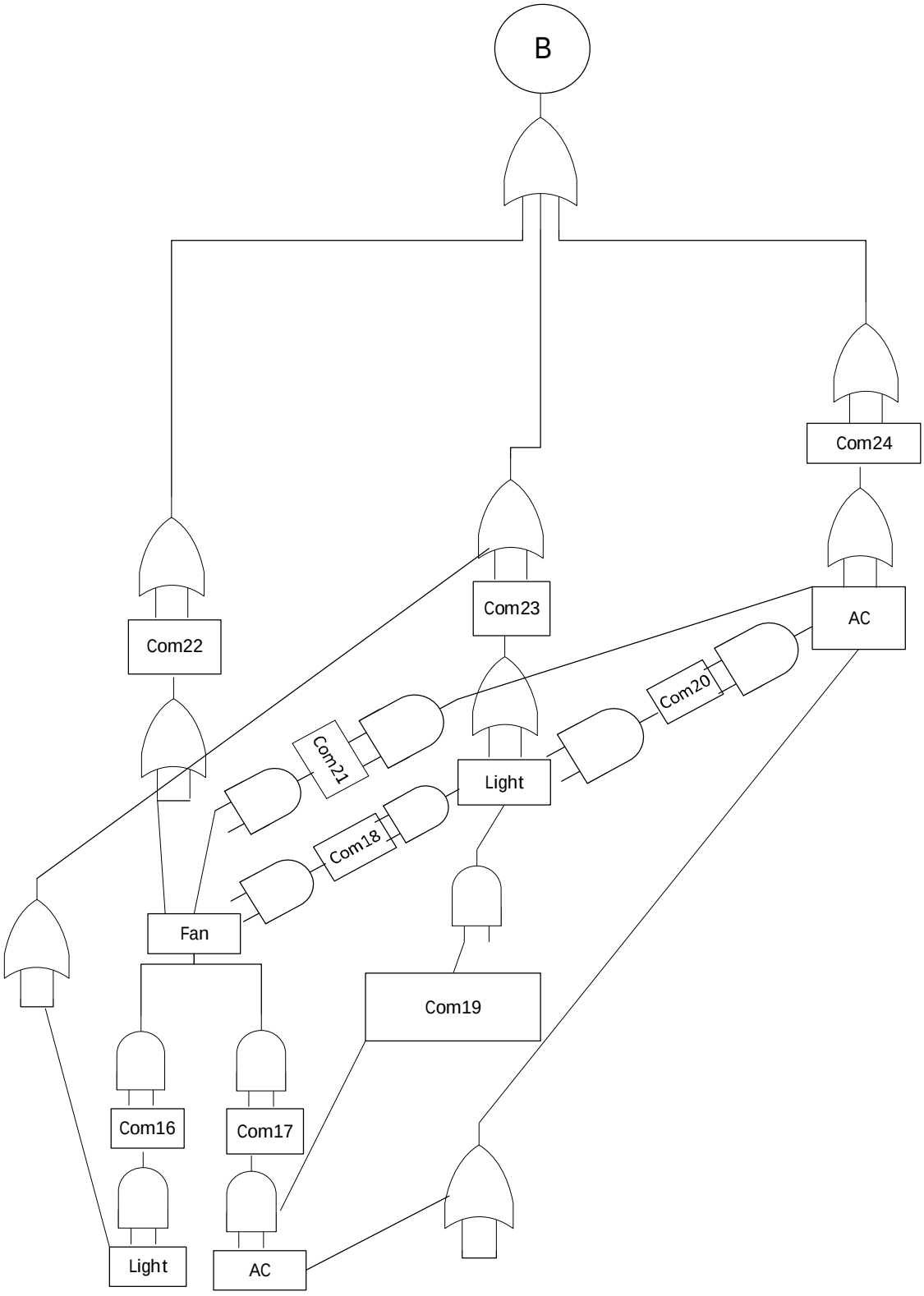


Figure 2: FT Diagram for the Prototype Model

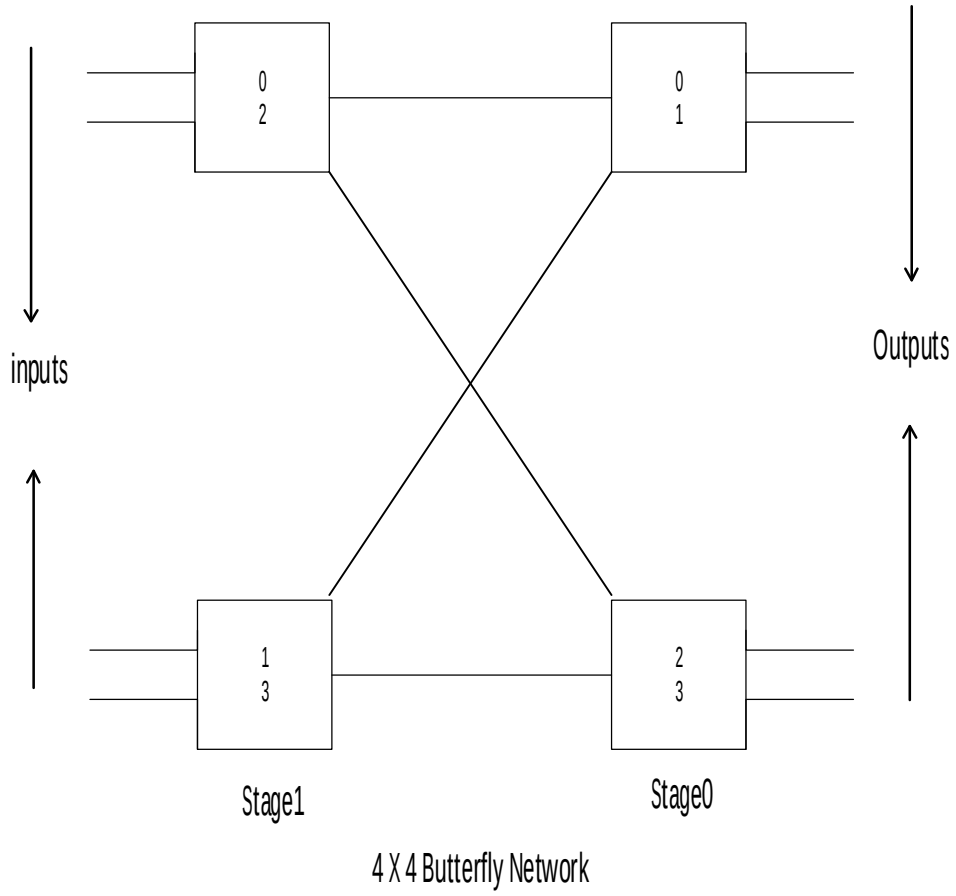


Figure 3: 4 X 4 Butterfly network

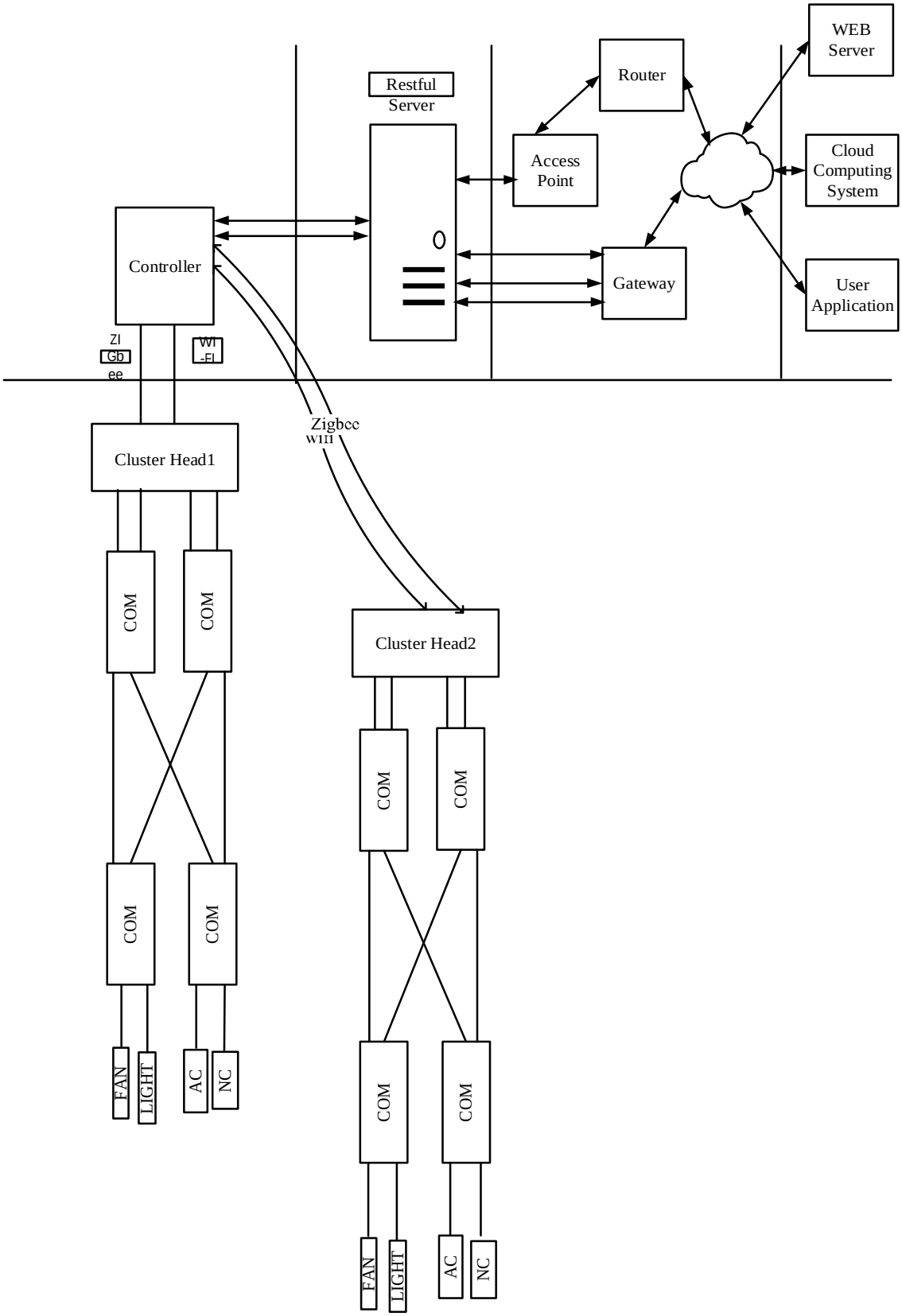


Figure 4: Modified IoT network using Butterfly topolo**gy**

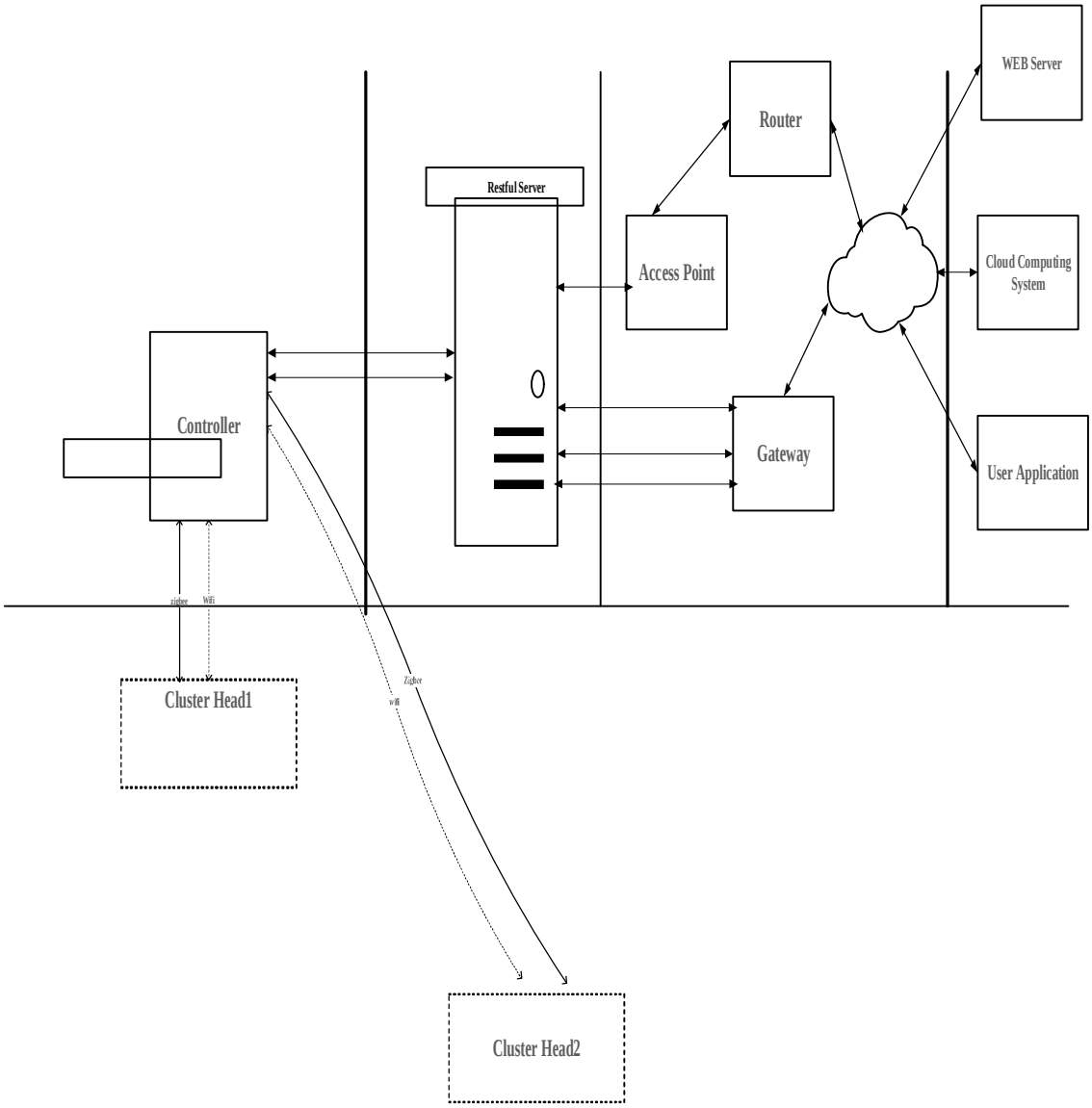


Figure 5: Modified IoT Network from Cluster Head

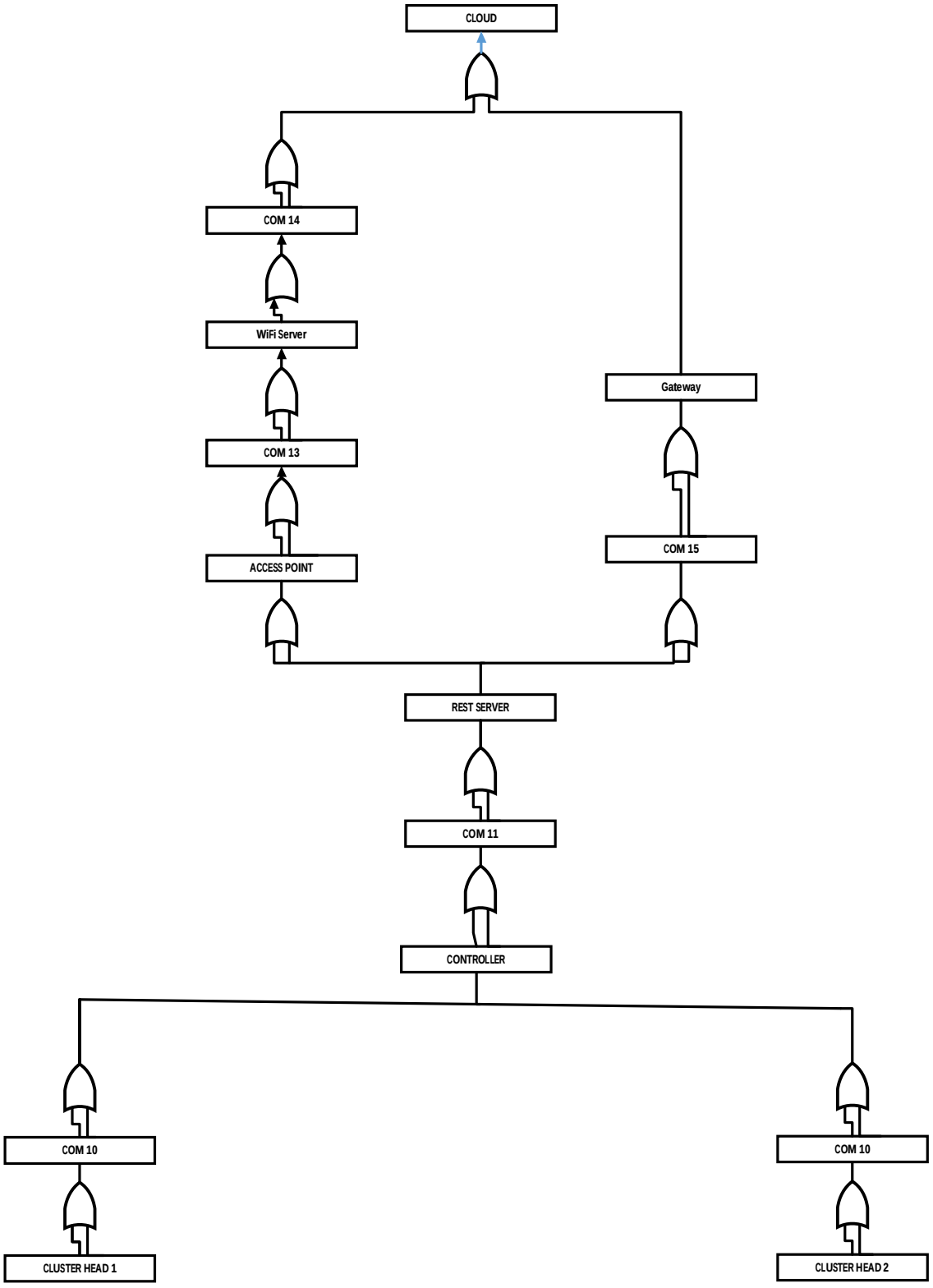


Figure 6: FTA Diagram for Modified IoT network

Table 1: Comparison of Fault Tolerance approaches

Sno.	Author Name	Network Failures											Distributed Networks					Wireless Networks					Fault Tolerance – Cloud Computing		
							Node Failures																		
		Frameworks	Architectures	Topologies	Link Failures	Path Failures	Power depletion	Energy depletion	Mis-Behaving nodes (Increase in Latency)	Faulty Nodes	Communication/ Messaging Failures\Routing	Device failures based on health indexing and functional relationships among the devices	Network Management	Scaling	Heterogeneity	Real-time	Parallel Processing	Power Depletion	Environmental Impact	Radio Interference	Asymmetric Communication Links	Dislocation of Sensor Nodes	Collision	Network-level	Service Level	Migration Level
1.	Maheswari				√		√	√	√	√			√													
2.	Paul Rubel													√	√	√	√									
3.	Sylvain Cherrier										√															
4.	Chen Wang1											√														
5.	Gholamreza Kakamanshadi																	√	√	√	√	√	√			
6.	DBK Kamesh				√	√			√	√																
7.	Susmitha																							√		
8.	Kai Fan			Random																						

Table 2: Fault computations based on FT Diagram

Sl.no	Device	Success Rate	Gates used For Connection	Preceding Devices					Combined Success Rate
				Device name D1	Device name D2	Device name D3	Device name D4	Device name D5	
				Success Rate S1	Success Rate S2	Success Rate S3	Success Rate S4	Success Rate S5	
1	LIGHT	0.980	AND	COM4	COM3				0.810
				0.900	0.900				
2	COM1	0.900	AND	LIGHT					0.810
				0.810					
3	FAN	0.980	AND	COM1	COM2				0.729
				0.810	0.900				
4	AC	0.980	AND	COM5	COM6				0.810
				0.900	0.900				
5	COM2	0.900	AND	AC					0.810
				0.810					
6	COM3	0.900	AND	FAN					0.729
				0.729					
7	COM4	0.900	AND	AC					0.810
				0.810					
8	COM7	0.900	OR	FAN					0.729
				0.729					
9	COM5	0.900	AND	LIGHT					0.810
				0.810					
				0.810					
36	CLOUD	0.980	OR	COM14	GATEWAY				0.729
				0.729	0.729				

Table 3: Fault computations of the revised networking of prototype model

Sl.no	Device	Success Rate	Gates used For Connection	Preceding Devices					Combined Success Rate
				Device name D1	Device name D2	Device name D3	Device name D4	Device name D5	
				Success Rate S1	Success Rate S2	Success Rate S3	Success Rate S4	Success Rate S5	
1	Cluster Head1	0.998							0.998
2	Cluster Head2	0.998							0.998
3	COM5	0.900	OR	CLUSTER HEAD1					0.900
				0.998					
4	COM6	0.900	OR	CLUSTER HEAD2					0.900
				0.998					
5.	CONTROLLER	0.980	OR	COM5	COM6				0.900
				0.900	0.900				
6.	COM7	0.900	OR	CONTROLLER					0.900
				0.900					
8	COM8	0.900	OR	REST SERVER					0.900
				0.900					
9	ACCESS POINT	0.980	OR	COM8					0.900
				0.900					
				0.810					
15	CLOUD	0.980	OR	GATEWAY	COM10				0.900
				0.900	0.900				