Journal of
Sensor and Actuator Networks

MDPI

Article

Enhancing the Fault Tolerance of a Multi-Layered IoT Network through Rectangular and Interstitial Mesh in the Gateway Layer

Sastry Kodanda Rama Jammalamadaka [1,*], Bhupati Chokara [1], Sasi Bhanu Jammalamadaka [2], Balakrishna Kamesh Duvvuri [3] and Rajarao Budaraju [1]

[1] Department of IoT and ECS, K L Deemed to be University, Vaddeswaram 522501, India; bhupati@kluniversity.in (B.C.); rajaraob@yahoo.com (R.B.)
[2] Department of Computer Science and Engineering, CMR College of Engineering and Technology, Hyderbad 501401, India; bhanukamesh1@gmail.com
[3] Department of Computer Science and Engineering, Mallareddy Engineering College for Women, Secunderabad 500100, India; kameshdbk@gmail.com
* Correspondence: drsastry@kluniversity.in

Abstract: Most IoT systems designed for the implementation of mission-critical systems are multi-layered. Much of the computing is done in the service and gateway layers. The gateway layer connects the internal section of the IoT to the cloud through the Internet. The failure of any node between the servers and the gateways will isolate the entire network, leading to zero tolerance. The service and gateway layers must be connected using networking topologies to yield 100% fault tolerance. The empirical formulation of the model chosen to connect the service's servers to the gateways through routers is required to compute the fault tolerance of the network. A rectangular and interstitial mesh have been proposed in this paper to connect the service servers to the gateways through the servers, which yields 0.999 fault tolerance of the IoT network. Also provided is an empirical approach to computing the IoT network's fault tolerance. A rectangular and interstitial mesh have been implemented in the network's gateway layer, increasing the IoT network's ability to tolerate faults by 11%.

Keywords: IoT networks; service layer; gateway layer; rectangular and interstitial mesh networks; computing fault tolerance

Citation: Jammalamadaka, S.K.R.; Chokara, B.; Jammalamadaka, S.B.; Duvvuri, B.K.; Budaraju, R. Enhancing the Fault Tolerance of a Multi-Layered IoT Network through Rectangular and Interstitial Mesh in the Gateway Layer. *J. Sens. Actuator Netw.* **2023**, *12*, 76. https://doi.org/10.3390/jsan12050076

Academic Editors: Tee-Hang Meen, Charles Tijus, Cheng-Chien Kuo, Kuei-Shu Hsu, Kuo-Kuang Fan, Jih-Fu Tu and Lei Shu

Received: 31 July 2023
Revised: 29 September 2023
Accepted: 10 October 2023
Published: 16 October 2023

1. Introduction

In an IoT network, many devices, such as sensors, actuators, servers, gateways, controllers, base stations, etc. [1], communicate as per the established communication system. IoT networks range from small and local to remotely connected global systems [2]. Many small devices (nanoscale) and newer communication technologies [3] are being developed and used daily.

Smaller devices are evolving rapidly, and millions of devices are connected to the cloud through the Internet. The direct connection of sensors/actuators to the cloud requires minimal infrastructure. However, implementing such a system leads to many bottlenecks, including latency, the speed of data transmission, data handling capabilities and proprietary protocols.

The computation, storage and network resources are brought closer to the IoT devices through the implementation of the edge/fog computing method, which has helped in eliminating the bottlenecks involved when sensors directly communicate with the cloud through the Internet (Pan and McElhannon, 2018 [4]; Adhinugraha et al., 2020 [5]).

A gateway connects the local devices to the Internet through routers wherever required. A gateway helps small devices to connect to the Internet, thus avoiding the necessity of

connecting the small devices directly to the Internet. A gateway allows small devices in an IoT network to be connected to traditional networks [6,7]. A gateway resembles a relay point to transmit data from a local IoT network to a global network.

The way in which small devices are connected to the gateway through the routers, and the way in which the devices communicate through the gateway, is a significant issue. The networking topology used to connect the devices to the gateway remains a significant challenge. Instead of improving or preserving the IoT network's fault tolerance, most of the topologies discussed in the literature have focused on cost reduction. The number of paths that are eliminated when a fault happens depends on the routing topology [8]. It takes time to reconstruct the networking topology when a fault happens, causing a delay in the response time. This is unacceptable when the IoT system is implemented in a real-time system.

Several gateways may have to be used to handle the traffic. The use of multiple gateways ensures that there is no single point of failure (Liu et al., 2017 [9]; Sahni et al., 2017 [10]). An IoT network's reliability is also enhanced when multiple gateways are used. The network traffic is expected to divert to another gateway when a gateway fails. Network broadcasting is generally used to find an alternative gateway (Tanenbaum and Wetherall, 2011 [11]).

When many gateways fail at once, broadcasting the message to all the nodes will block the entire network. Locating a replacement gateway many hops away increases the communication latency (Lin et al., 2018 [12]). A Voronoi model can be used to determine the distance coverage to a gateway (Liu et al., 2021 [13]; Zhu et al., 2012 [14]).

Several networking topologies can be used to establish a network between the service's servers, routers and gateways. A mesh network topology reduces network congestion in the cloud (Safar M et al. [15]). A mesh networking topology also helps to locate the processing, storage or analytical services near the location from which data are routed to the cloud, thereby reducing network traffic (Tran Q.T. et al. [16]). In the case of mesh networks, sensors/servers are connected to the routers and the routers to the gateways. The data are processed at the devices (servers, routers and gateways). These devices are used not only for computing but also for routing. A mesh network is reliable, and there is no single point of failure (Xuan K et al. [17]).

Numerous routers exist in a mesh network, which can be used for the transmission of the data. A mesh network can be easily expanded by adding processing units, including servers, routers and gateways. The multi-hop approach can cover larger areas of processing devices (Kolahdouzan M et al. [18]). The devices formed into a mesh network need not be equidistant. The devices are located such that they are within the broadcasting range. They should be routed from the sensing/servicing devices so that data are moved to one of the gateways in the network. When any gateway is out of order, other operational gateways must share the load.

A Voronoi diagram can be used to partition the processing nodes based on the nearest neighbor concept (Cheema M.A. et al. [19], Yang S et al. [20]). A Voronoi diagram (VD) represents a road network with weights attached to the roots, instead of considering the Euclidean distances between the devices. A network graph can be developed considering the relationships between the computing resources with the help of a network Voronoi diagram (NVD) (Shao Z., Taniar D et al. [21], Lee I et al. [22], Aurenhammer F et al. [23]). A higher-order Voronoi diagram can be constructed (HOVD) by segregating the cells in the network into different regions and then connecting the regions (KP Gummadi et al. [24], Okabe A et al. [25]). A reverse k-nearest neighbor method (RkNN) can also be used to regroup the cells (Taniar D et al. [26]).

Most of the studies presented in the literature involve connecting to the nearest neighbor node as fast as possible so that communication progresses in the event of the failure of one or more processing nodes. None have shown the extent to which the fault tolerance of the network is improved in quantitative terms.

Modern IoT networks are built using multiple layers, including devices, controllers, services, gateways and the cloud. Each layer implements a different networking topology.

In addition to computing the fault tolerance of a specific layer, there is a need to compute the overall fault tolerance of the IoT network. The processing is done in the service layer. The use of mesh networks to connect the service's servers to the gateways has yet to be discussed. A pragmatic method of computing the fault tolerance of the mesh network considering the service's servers, routers and gateways still needs to be developed.

A mesh network topology is a widely used topology in IoT networks. However, mesh networks are used to achieve certain functionalities and are not focused on enhancing the IoT network's fault tolerance. A rectangular and interstitial mesh network topology exists, and very few use these networks to build functionality. However, we have yet to see a contribution that includes the usage of a rectangular and interstitial mesh network topology in the gateway layer to enhance the fault tolerance of the IoT network. Mission-critical applications require fault-tolerant IoT networks, catering to any failures, whether they belong to hardware, software or network failures.

The deployment of IoT nodes is dependent on the application requirements. Here, we refer to highly tolerant IoT systems, primarily required for building mission-critical systems relating to aerospace, defence and process engineering. Many IoT-based applications use a rectangular and interstitial mesh network topology. They are mainly related to defence and aerospace. The fault tolerance of the IoT network could be enhanced by introducing the rectangular and interstitial mesh network topology in the gateway layer, as presented in this paper. We will describe the types of functions implemented in the gateway layer and why they must be highly fault-tolerant. When using the rectangular and interstitial mesh network in the gateway layer, it is not yet possible to compute the fault tolerance of the IoT network using probabilistic evaluation.

Challenges and Motivation for the Research

Enhancing the IoT network's fault tolerance at the gateway level is a challenge since this vital junction point frequently experiences problems that place the network's overall functionality at risk. Finding the least expensive solution is the greatest challenge. We are driven to investigate these topics because of our numerous interactions with defence aerospace personnel.

2. Related Work

T. Saha et al. [27] proposed a gateway that acts as an interface between the Internet and the rest of the IoT network. The gateway is designed to deal with hardware, software and connection failures and the overall load balance of the network. Their proposed framework involves a set of observers connected with the gateway and sensors connected with the observers. The framework built into the observers comprises prevention and detection algorithms to prevent communication failures between the sensor nodes and gateway, provide alternative reliable transmission paths and detect node faults in the early stages. They have yet to reveal how much adopting the suggested models will increase the IoT network's capability for fault tolerance.

A technique to identify the communication routes between nodes in a wireless sensor network (WSN) was put forward by Ma et al. [28]. However, their approach presupposes a setting where data are aggregated from the sensor nodes to the sink nodes. Instead of consolidating the data into a single node, the IoT sensor network expects the distributed resources to function together. The topology uses a multi-routing tree to ensure fault tolerance and reduce power usage.

A resource allocation approach for a mobile network environment was proposed by Ismail et al. [29]. They employ a system that involves locating the closest access point to communicate. A mobile device is capable of concurrent connections to numerous access points. This is in contrast to the IoT environment, where communication is carried out by choosing one of the candidate's gateways.

Li et al. [30] investigated the use of a genetic algorithm to build a routing tree for a P2P network in distributed interactive applications. Their approach does not consider

fault tolerance and instead tries to optimize the communication speed between two nodes in the ecosystem.

Karthikeya et al. [31] suggested a technique to locate the gateway in the best possible location. By strategically placing the gateway, their strategy can reduce the expense of introducing it into the environment.

Kim et al. [32] researched a method to solve the resource allocation problem in the gateway in the IoT environment. Their method uses a genetic algorithm to find the optimum path from a source node to a destination node. This method does not aim at quantifying the improvement in a specific layer of an IoT network.

Takahashi, R. [33] has proposed a method to generate a fault-tolerant networking and routing method using a genetic algorithm such that all the communication paths do not concentrate on one gateway.

Leonardi et al., 2018 [34] and Rondón et al., 2019 [35] have proposed a mesh network for massive device deployment to ensure no single point of failure. They have not discussed fault rate computations considering the mesh networks.

He et al., 2007 [36] and Seyedzadegan et al., 2013 [37] have presented gateway management techniques, such as optimal gateway placement, to handle massive devices and high traffic demands within an IoT network. Shih and Wu, 2016 [38] have presented multi-protocol gateways to support heterogeneous networks. They have yet to focus on the issue of fault tolerance.

Jean-Philippe et al., 2004 [39] have shown that mesh networks are also prone to both unplanned blackouts (natural disasters, overload, wearout, bug attacks on devices) and planned blackouts (hardware replacements and maintenance). They have suggested that a gateway recovery strategy is required to maintain network reliability.

In order to implement loopback methods (Médard and others (2002) [40]), Choi et al. (2004) [41] provided pre-computed backup paths that should be employed in the event of a network failure.

To reduce traffic strain on a single gateway, the usage of several gateways has been suggested by Lakshmanan et al. (2009) [42] and Kawai et al. (2014) [43].

Using an order-k HVD, Adhinugraha et al., 2021 [44] have supplied pre-computed backup routers. These pieces were all centered around the failure of a single entrance. The problem of multiple chained gateway failure has yet to be considered.

In the first approach, the reserved resources are only used if a network event occurs. In contrast to pre-computed resources, the second approach does not require reserved or pre-computed resources because the alternative plan is created by integrating a particular recovery algorithm into the network after the incident. Reducing the allocation of irrational resources is the key benefit of computing the recovery plan on the fly. The resources required for a backup plan determine the cost of reserving resources.

An automatic recovery system (ARS) was suggested by Kim and Shin (2011) [45] so that a wireless mesh network (WMN) could automatically recover from local link failures and maintain network availability.

Effective routes can be found using route calculations such as SDNMesh and RADAR, proposed by Gilani et al. in 2020 [46] and Sarkar et al. in 2007 [47], respectively.

Every gateway will take a modest hit thanks to the traffic handling strategy provided by Mahiddin and Sarkar (2019) [48]. In the event that the primary gateway breaks, Minh et al. (2014) [49] suggested using an OEMAN technique to designate one mobile device as a backup access point.

To compute k backup gateways and assess various gateway diversion strategies to direct a router output to a gateway with the fewest hops, Kiki Adhinugraha [50] presented a generalized model based on the k-hops Voronoi diagram.

A distribution approach using a hops Voronoi diagram (HVD) has been presented by Kiki Adhinugraha et al. [51]. The closest number of hops from the router is used to determine dependent routers. The strategies put forth in the literature that are designed to

increase or maintain the fault tolerance level of the Internet of Things network have been categorized by Moghaddam et al. [52].

3. Research Gap and Research Objectives

3.1. Research Gap

Many have proposed improvements in the device, base station, controller and service layer to enhance the IoT network's fault tolerance Level. However, the greatest challenge is ensuring smooth operation in the gateway layer, as it makes the application non-operational if any component fails. The use of a single gateway carries a high level of risk. Many gateways require a complex network, which makes the cost of IoT networks very high. There is a need to implement a low-cost networking solution in the gateway layer so that the IoT network is fully operational and the fault tolerance level of the IoT network is enhanced.

3.2. Research Objectives

1. Enhance the fault tolerance of the IoT network by introducing a rectangular and interstitial mesh network topology;
2. Develop an improved probabilistic formulation that can compute the fault tolerance of the rectangular and interstitial mesh network topology;
3. Develop two algorithms that help to convert an IoT network into a fault tree and generate a failure rate computation table given a fault tree;
4. Develop a computational method that can be used to compute the fault tolerance of the entire IoT network;
5. Implement a failure analysis model to justify the efficiency of using the rectangular and interstitial mesh network topology over other techniques;
6. Compute and compare the results through empirical models and external observations.

4. The Overall Method to Enhance the Fault Tolerance of the IoT Network with Changes Induced in the Gateway Layer

The flow diagram for the implemention of the suggested methods is shown in Figure 1. Bhupathi et al. outlined the metrics, prototype IoT network, fault tree analysis, crossbar network, detection and isolation of faults within sensing and actuating devices to prevent potential fault injection in the device layer, generation of an FTA model and enhancement of the fault tolerance level of an IoT network [53,54].

With the addition of a second base station, the IoT network's controller layer has undergone more improvements. A network of relays with redundancy introduced has been used to connect the cluster heads and the second base station, to overcome the risk of failure of the first base station. Each relay has a sophisticated pathfinding algorithm installed to discover the quickest route with the least amount of traffic and significantly reduce the latency [55].

Section 8.1 compares gateway techniques to achieve improvements in the IoT network's fault tolerance.

Implementing the load balancing feature within the controllers, which are networked by an I2C network, has further strengthened the capability of the IoT network to survive during the occurrence of faults. A smart service to estimate missing data has been introduced to the service servers.

The IoT network is further improved by adding a rectangular and interstitial mesh network in the gateway layer to improve the fault tolerance in this network layer. A new reliability model has been proposed to compute the fault tolerance considering a rectangular interstitial mesh network.

A probability method is presented in this paper, which is used to compute the fault rate of the rectangular and interstitial mesh network introduced in the gateway layer of the network. The IoT diagram is modified by replacing the rectangular and interstitial mesh

network in the gateway layer, which is assigned with the fault rate of the rectangular and interstitial mesh network.

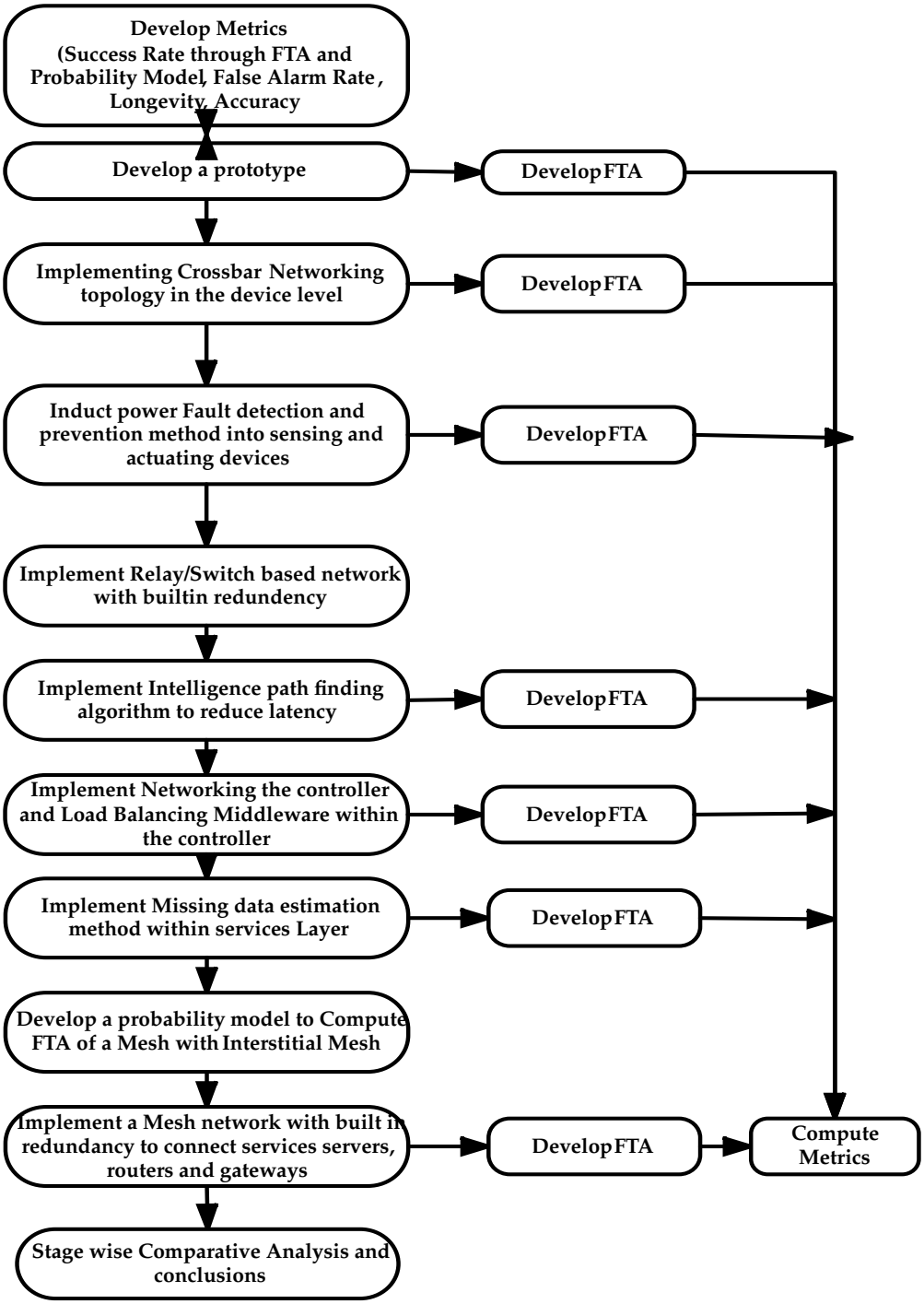

Figure 1. Overall computational strategy to enhance fault tolerance in the gateway layer.

The revised IoT network generates an FTA diagram using Algorithm 1, and a fault rate table is generated considering the FTA diagram using Algorithm 2. The fault rate of the IoT network is the fault rate of the root node of the network.

A fault analysis model has been presented for comparison with other methods, of which the success rate is 99.99, while the success rates of other methods when induced in the gateway layer are negligible. Other methods offer the least resistance to failure when the number of node failures is >=4.

Algorithm 1: Generating Fault Tree.	
Step 1	Identify the hardware hierarchy in an IoT network and update a database.
Step 2	Copy the IoT diagram's clusters, transform them into hierarchical models and update the database's entries.
Step 3	Record the networking topologies utilized in the Internet of Things network, calculate the fault rate for each one using the relevant probability model and add a database element to reflect the computed fault rate.
Step 4	Update the database with manufacturers' information about other devices' failure rates.
Step 5	Update the database after recording the relationship (OR, AND) between each device and its ancestors.
Step 6	Create a graph model from the linear tree.

Algorithm 2: Generating Fault Rate Table.	
Step 1:	Query the database's elements according to the preceding relationships that connect the child nodes in hierarchical order.
Step 2:	Determine the fault rate of the outgoing device using AND/OR rules.
Step 3:	Multiply an outgoing device's fault rate by the incoming device's fault rate.
Step 4:	The outgoing device's fault rate is the lowest of the incoming devices' fault rates if the relationship between the devices is an OR relationship.
Step 5:	Determine the root device's fault rate. No parents exist for a root device.
Step 6:	Create a fault computation table.

5. Example IoT Network

Table 1 shows various methods that help to enhance the fault tolerance in different network layers of an IoT network. An example IoT network is shown in Figure 2. The example IoT network implements various fault tolerance enhancing methods in different network layers, detailed in Table 1.

Table 1. Implementation in the sample IoT network in the service layer.

Layer	Innovation Implemented	Reference
FTA Computation Model	A hybrid model that helps to compute the FTA considering a linear and probability model	[53]
Device Layer	Predicting the occurrence of a power fault and mitigating it through an isolation procedure	[54]
	Using a crossbar network in the device layer to achieve a seamless connection between the devices and the cluster heads	
Controller Layer	Connecting the cluster heads to the first base station using a peer-to-peer network	[55]
	Connecting the second base station to the cluster heads through a distributed relay network	
	A new algorithm to find the shortest path to connect a cluster head to a second base station	
	Networking controllers and implementing load balancing within the controllers	
	Connecting the controller to the servers of the service through a crossbar network	
Service Layer	Implementing a machine learning-based missing data estimation model	

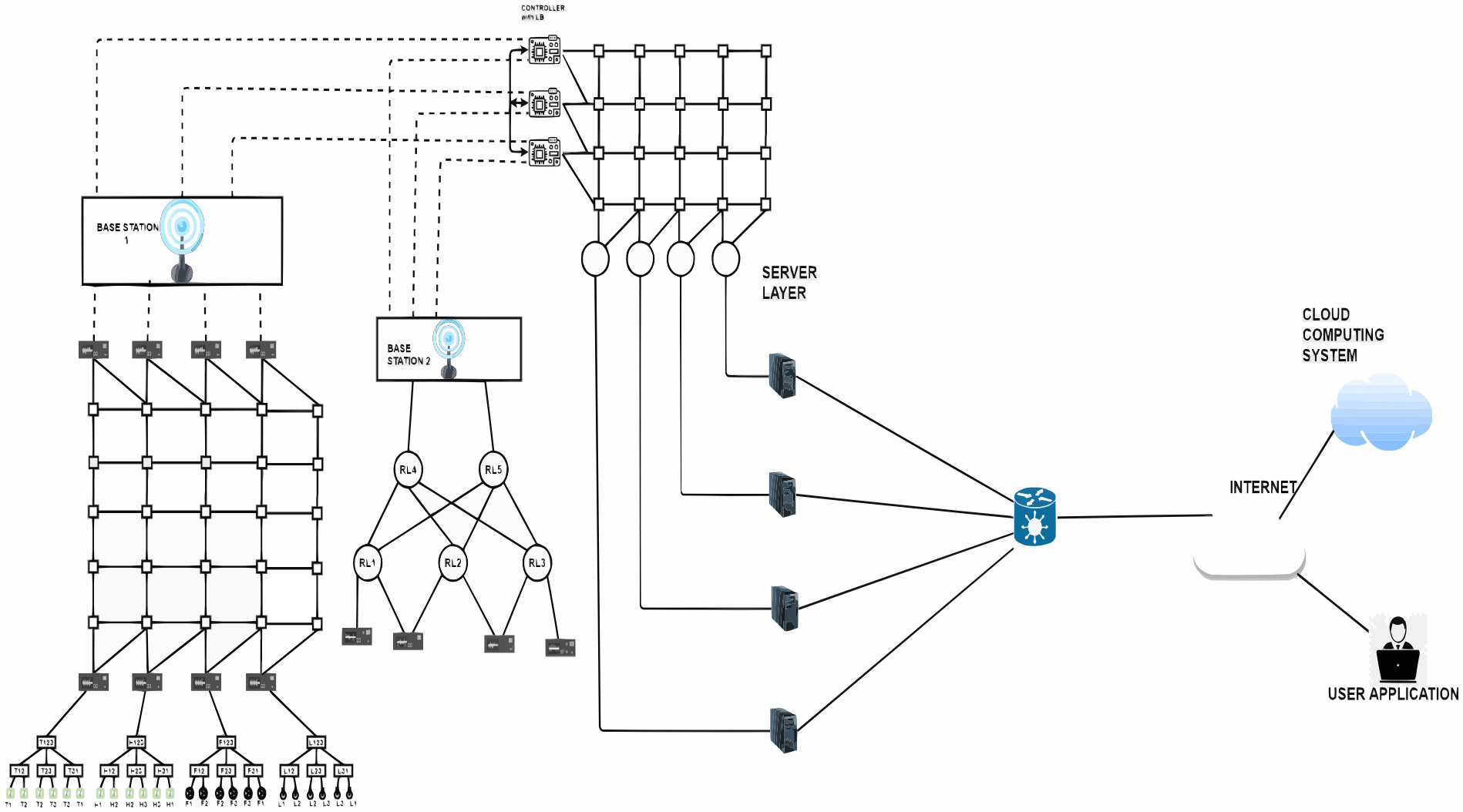

Figure 2. Example IoT network.

The sensors and actuators are connected to the cluster heads considering the availability of redundancy, which is manifested as the linearization of the sensors. The cluster heads are connected to two base stations through a parallel channel and a separate relay network. Three microcontrollers are connected to both the base stations to provide redundant communication channels.

An I2C network connects the microcontrollers, allowing for connectivity issues between the controllers and the crossbar network. A crossbar network links the controllers to the service servers. The service server takes requests from clients or users, runs code relating to the requested services and sends the results to the controllers or the client. A new service is developed in the service server to determine whether the missing data will not be communicated due to a device failure in the layer.

A single device replaces the crossbar network at the device and controller levels, and probability models coupled to crossbar networks construct a failure tree of the sample IoT network. Maintaining connectivity between the service servers, routers and gateways requires much work. This layer handles a large proportion of the data flow, and device failures in this layer severely limit the IoT network's ability to tolerate faults. The fault tree and fault rate computation tables are produced using Algorithms 1 and 2.

6. Investigations and Findings

6.1. Success Rate Computation Method for Rectangular and Interstitial Mesh Network

In a mesh network, all nodes are pressing nodes. The network can be formed using the service's servers, routers and gateways. All the non-boundary nodes have 4 incident links. To send a message from a node to a node that is not a neighbor, a path from the node from which the message is initiated to a destination node must be identified. The message must be forwarded involving the intermediate nodes along the path. A mesh network loses its property that there should be four links to communicate from a node when it breaks down for any reason. To provide tolerance, redundant nodes are added. The redundant nodes are switched in when any of their neighboring nodes fails.

A mesh network involving four service servers, four gateways and 10 routers is shown in Figure 3. Four routers are spare, and they will come into force only when a neighboring node fails.

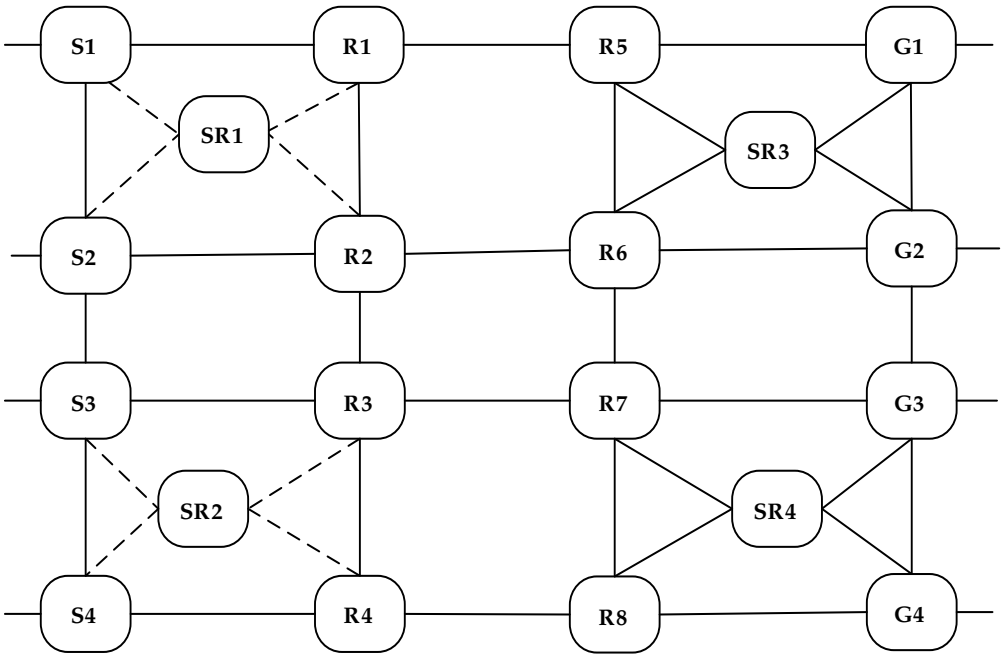

Figure 3. Mesh network connecting service's servers to the gateway. S1, S2, S3, S4 = servers. SR1, SR2, SR3, SR4 = spare routers. G1, G2, G3, G4 = gateways. R1–R8 = routers.

The reliability of a mesh network is the probability that the mesh property is retained. Using Equation (1), the fault tolerance can be computed.

R = the reliability or the success rate of servers, routers or gateways = 0.98;

N = number of rows in the network = 4;

M = number of columns in the network = 4;

C = number of clusters in the network = (N ∗ M)/4, with each cluster formed using 4 primary nodes and a spare node = 4;

n = number of sub-meshes row-wise = 2;

m = number of sub-meshes column-wise = 2;

k = number of possible allocations = N/n ∗ M/m = 4, which leads to a 1-of-k model. The reliability of such a model can be computed using Equation (1).

S = probability that an n ∗ m sub-mesh can be allocated.

$$S = (1 - R^{nm})^k = (1 - 0.98^4)^4 = 0.99 \quad (1)$$

6.2. Algorithm to Convert an IoT Network into an FTA Graph

To create linear IoT models, device clusters in an IoT network must first be transformed into linear models. The remaining portion of the network is integrated with the linear models related to the clusters to create a complete linear network. Algorithm 1 creates an FTA graph from an IoT network.

6.3. Algorithm to Convert an FTA Graph into a Table of Fault Rate Computations

The failure rate of each device and the network is calculated using a linear tree, the specifics of which are saved in a database, and a fault rate table is created using Algorithm 2. The program analyzes the connections between the devices, using AND/OR operations, and calculates each device's fault rate until the root node is located.

For brevity, the defect rates for the devices utilized in the IoT network are obtained from the relevant manufacturers and calculated using device-specific empirical formulae. Through gates, the outgoing and incoming devices are linked. The AND/OR gates control the connectivity of the device, the precedence rule and the fault rates.

6.4. Revised IoT Network

The network gateway layer updates include the following:

1. A mesh network is established connecting the service servers, routers and gateways;
2. One spare router is included for every four processing nodes;
3. All the gateways are connected to the cloud through the Internet;
4. The gateway is loaded with intelligent software for receiving, buffering, de-buffering, selecting communication speeds and implementing the accounting systems.

The revised network is shown in Figure 4, and the linearized IoT diagram using Algorithm 1 is shown in Figure 5.

Routers are expensive. A less expensive topology is required while simultaneously providing the redundancy to enhance the fault tolerance. This is where the rectangular and interstitial mesh network topology is introduced.

We have used only a 4 × 4 rectangular and interstitial mesh network, which is very simple to implement. More connections are required. Experimentation is also easy. If the workload is smaller, we can use a 2 × 2 network, in which case the number of servers used in the service layer would be smaller. The physical failure has been counted externally and compared with empirical computations. A simulation of working rectangular and interstitial mesh networks is also done.

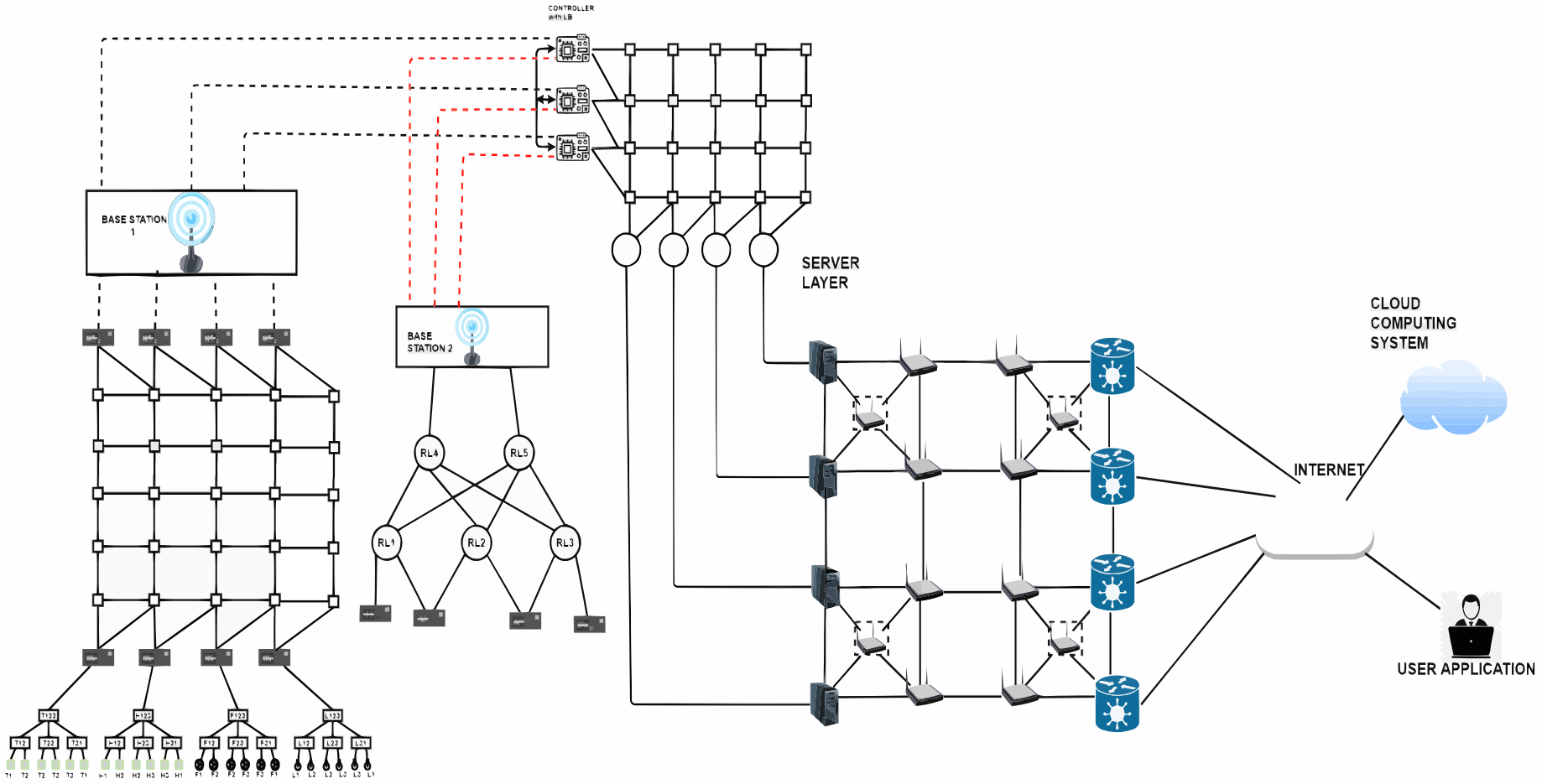

Figure 4. Revised IoT network with revisions in the gateway level.

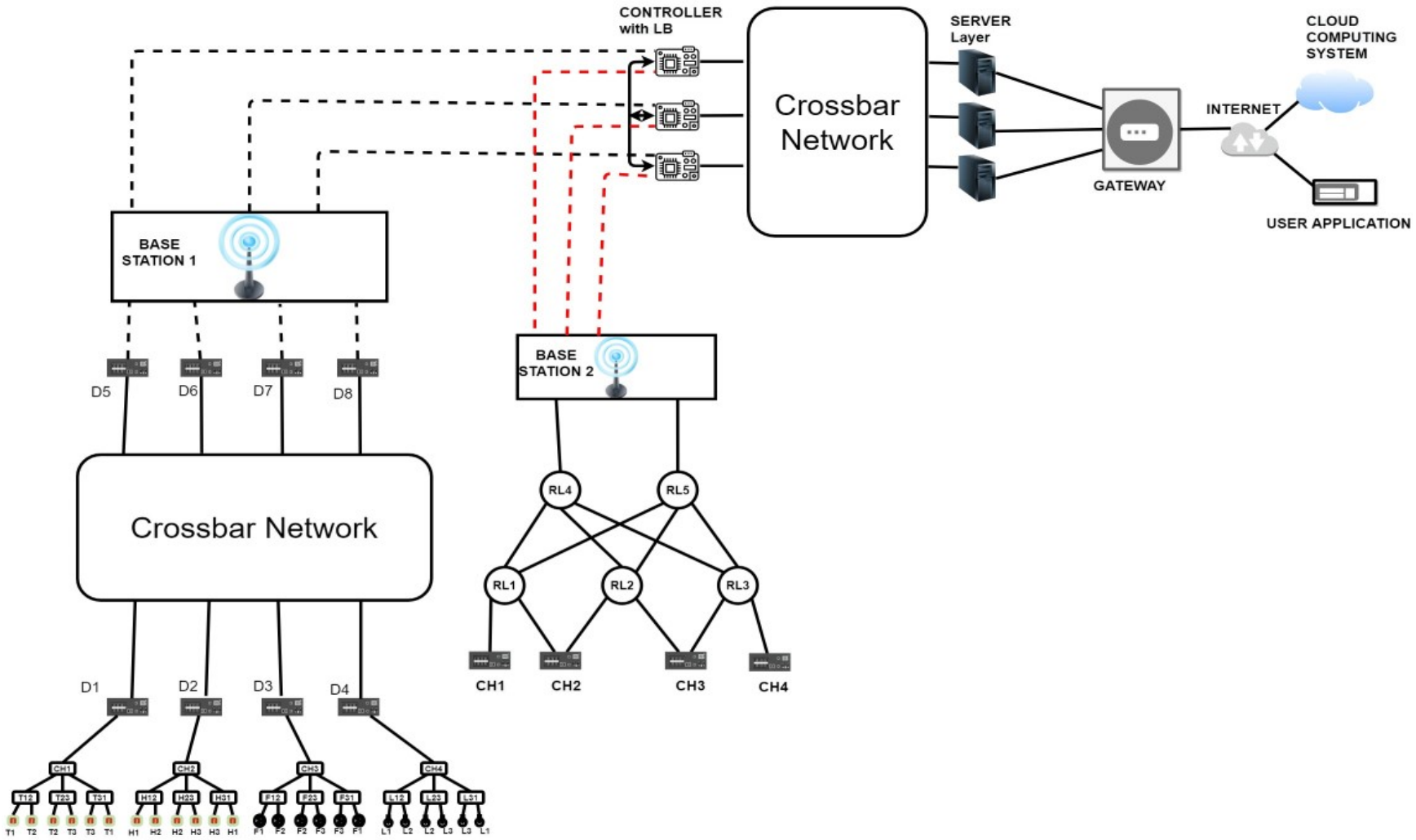

Figure 5. Linearized example IoT network.

7. Experimentation and Results

7.1. FTA, Example, IoT Network

Algorithm 1 creates an FTA diagram, as shown in Figure 6.

A single device replaces every complex networking topology, and then an FTA graph is generated considering the connectivity among all the devices, which is then used to compute the overall fault tolerance of the IoT network.

7.2. Success Rate Computation of Example IoT Network

Algorithm 2 is used to compute each device's success rates and consider the logical and precedence relations among the devices. A table displaying the fault rate computations is produced. Table 2 shows 0.9800 as the fault tolerance of the example IoT network.

Table 2. Success rate computations of an example IoT network.

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	
				Success Rate S1	Success Rate S2	Success Rate S3	Success Rate S4	
1	Temp-Sensor-1	0.95						0.950
2	Temp-Sensor-2	0.95						0.950
3	Temp-Sensor-3	0.95						0.950
4	T12-Dummy	0.95	OR	T1 0.950	T2 0.950			0.950
5	T23-Dummy	0.95	OR	T2 0.950	T3 0.950			0.950
6	T13-Dummy	0.95	OR	T1 0.950	T3 0.950			0.950
7	T-123	0.95	OR	T12 0.950	T23 0.950	T31 0.950		0.950
8	Humidity- Sensor-1	0.95						0.950
9	Humidity- Sensor-2	0.95						0.950
10	Humidity- Sensor-3	0.95						0.950
11	H12-Dummy	0.95	OR	H1 0.950	H2 0.950			0.950
12	H23-Dummy	0.95	OR	H2 0.950	H3 0.950			0.950
13	H31-Dummy	0.95	OR	H3 0.950	H1 0.950			0.950
14	H-123	0.95	OR	H12 0.950	H23 0.950	H31 0.950		0.950
15	FAN-1	0.95						0.950
16	FAN-2	0.95						0.950
17	FAN-3	0.95						0.950
18	F12-Dummy	0.95	OR	F1 0.950	F2 0.950			0.950
19	F23-Dummy	0.95	OR	F2 0.950	F3 0.950			0.950
20	F31-Dummy	0.95	OR	F3 0.950	F1 0.950			0.950
21	F-123	0.95	OR	F12 0.950	F23 0.950	F13 0.950		0.950
22	Light 1	0.95						0.950
23	Light 2	0.95						0.950
24	Light 3	0.95						0.950
25	L12-Dummy	0.95	OR	L1 0.950	L2 0.950			0.950
26	L23-Dummy	0.95	OR	L2 0.950	L3 0.950			0.950
27	L31-Dummy	0.95	OR	L3 0.950	L1 0.950			0.950

Table 2. *Cont.*

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	
				Success Rate S1	Success Rate S2	Success Rate S3	Success Rate S4	
28	L-123	0.95	OR	L12 0.950	L23 0.950	L31 0.950		0.950
29	D1	0.95	OR	T-123/CH1 0.950				0.95
30	D2	0.95	OR	H-123/CH2 0.950				0.95
31	D3	0.95	OR	F-123/CH3 0.950				0.95
32	D4	0.95	OR	L-123/CH4 0.950				0.95
33	Device Level Crossbar NW (DLCB)	0.987	OR	D1 0.950				0.987
34	Device Level Crossbar NW (DLCB)	0.987	OR	D2 0.950				0.987
35	Device Level Crossbar NW (DLCB)	0.987	OR	D3 0.950				0.987
36	Device Level Crossbar NW (DLCB)	0.987	OR	D4 0.950				0.987
37	D5	0.95	OR	DLCB 0.987				0.987
38	D6	0.95	OR	DLCB 0.987				0.987
39	D7	0.95	OR	DLCB 0.987				0.987
40	D8	0.95	OR	DLCB 0.987				0.987
41	BS1	0.95	OR	D5 0.987	D6 0.987	D7 0.987	D8 0.987	0.987
42	RL1	0.95	OR	Cluster Head1 0.950	Cluster Head2 0.950			0.95
43	RL2	0.95	OR	Cluster Head1 0.950	Cluster Head2 0.950	Cluster Head3 0.950	Cluster Head4 0.950	0.95
44	RL3	0.95	OR	Cluster Head3 0.950	Cluster Head4 0.950			0.95
45	RL4	0.95	OR	RL1 0.950	RL2 0.950			0.95
46	RL5	0.95	OR	RL1 0.950	RL2 0.950			0.95
47	BS2	0.95	OR	RL4 0.950	RL5 0.950			0.95
48	CONTROLLER 1	0.9	OR	BS1 0.987	BS2 0.950			0.987
49	CONTROLLER 2	0.9	OR	BS1 0.987	BS2 0.950			0.987
50	CONTROLLER 3	0.9	OR	BS1 0.987	BS2 0.950			0.987
51	CONTROLLER LEVEL CROSSBAR NW	0.97	CROSSBAR NW	CONTROLLER 1 0.987	CONTROLLER 2 0.987	CONTROLLER 3 0.987		0.987
52	SERVER 1	0.98	OR	CONTROLLER LEVEL CROSSBAR NW 0.987				0.987
53	SERVER 2	0.98	OR	CONTROLLER LEVEL CROSSBAR NW 0.987				0.987
54	SERVER 3	0.98	OR	CONTROLLER LEVEL CROSSBAR NW 0.987				0.987
55	GATEWAY	0.98	OR	SERVER 1 0.987	SERVER 2 0.987	SERVER 3 0.987		0.987
56	INTERNET	0.95	AND	GATEWAY 0.987				0.980

7.3. FTA for Revised IoT Network

A single device with success rates determined by its associated probability model has taken the position of the crossbar networks in the device layer, service layer and rectangular and interstitial mesh network in the gateway layer. Figures 7 and 8 display the linearized revised IoT network and its associated FTA graph produced by Algorithm 1.

7.4. Success Rate Computation for Revised IoT Network

The success table was produced using Algorithm 2 with the FTA diagram as input. Table 3 displays the success table that was generated. All devices' success rates are calculated based on their relationships with other devices regarding precedence. The success rate of the updated IoT network is 0.9900, as seen in the table. To determine how many date stamps are missing from the data stored on the cloud server, the error log and the fault rate are computed, and it is discovered that the success rate is 0.9910.

Table 3. Success rate computations of an example IoT network.

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	
				Success Rate S1	Success Rate S2	Success Rate S3	Success Rate S4	
1	Temp-Sensor-1	0.95						0.950
2	Temp-Sensor-2	0.95						0.950
3	Temp-Sensor-3	0.95						0.950
4	T12-Dummy	0.95	OR	T1 0.950	T2 0.950			0.950
5	T23-Dummy	0.95	OR	T2 0.950	T3 0.950			0.950
6	T13-Dummy	0.95	OR	T1 0.950	T3 0.950			0.950
7	T-123	0.95	OR	T12 0.950	T23 0.950	T31 0.950		0.950
8	Humidity- Sensor-1	0.95						0.950
9	Humidity- Sensor-2	0.95						0.950
10	Humidity- Sensor-3	0.95						0.950
11	H12-Dummy	0.95	OR	H1 0.950	H2 0.950			0.950
12	H23-Dummy	0.95	OR	H2 0.950	H3 0.950			0.950
13	H31-Dummy	0.95	OR	H3 0.950	H1 0.950			0.950
14	H-123	0.95	OR	H12 0.950	H23 0.950	H31 0.950		0.950
15	FAN-1	0.95						0.950
16	FAN-2	0.95						0.950
17	FAN-3	0.95						0.950
18	F12-Dummy	0.95	OR	F1 0.950	F2 0.950			0.950
19	F23-Dummy	0.95	OR	F2 0.950	F3 0.950			0.950
20	F31-Dummy	0.95	OR	F3 0.950	F1 0.950			0.950
21	F-123	0.95	OR	F12 0.950	F23 0.950	F13 0.950		0.950
22	Light 1	0.95						0.950
23	Light 2	0.95						0.950
24	Light 3	0.95						0.950
25	L12-Dummy	0.95	OR	L1 0.950	L2 0.950			0.950
26	L23-Dummy	0.95	OR	L2 0.950	L3 0.950			0.950
27	L31-Dummy	0.95	OR	L3 0.950	L1 0.950			0.950

Table 3. *Cont.*

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	
				Success Rate S1	Success Rate S2	Success Rate S3	Success Rate S4	
28	L-123	0.95	OR	L12 0.950	L23 0.950	L31 0.950		0.950
29	D1	0.95	OR	T-123/CH1 0.950				0.95
30	D2	0.95	OR	H-123/CH2 0.950				0.95
31	D3	0.95	OR	F-123/CH3 0.950				0.95
32	D4	0.95	OR	L-123/CH4 0.950				0.95
33	Device Level Crossbar NW (DLCB)	0.987	OR	D1 0.950				0.987
34	Device Level Crossbar NW (DLCB)	0.987	OR	D2 0.950				0.987
35	Device Level Crossbar NW (DLCB)	0.987	OR	D3 0.950				0.987
36	Device Level Crossbar NW (DLCB)	0.987	OR	D4 0.950				0.987
37	D5	0.95	OR	DLCB 0.987				0.987
38	D6	0.95	OR	DLCB 0.987				0.987
39	D7	0.95	OR	DLCB 0.987				0.987
40	D8	0.95	OR	DLCB 0.987				0.987
41	BS1	0.95	OR	D5 0.987	D6 0.987	D7 0.987	D8 0.987	0.987
42	RL1	0.95	OR	Cluster Head1 0.950	Cluster Head2 0.950			0.95
43	RL2	0.95	OR	Cluster Head1 0.950	Cluster Head2 0.950	Cluster Head3 0.950	Cluster Head4 0.950	0.95
44	RL3	0.95	OR	Cluster Head3 0.950	Cluster Head4 0.950			0.95
45	RL4	0.95	OR	RL1 0.950	RL2 0.950			0.95
46	RL5	0.95	OR	RL1 0.950	RL2 0.950			0.95
47	BS2	0.95	OR	RL4 0.950	RL5 0.950			0.95
48	CONTROLLER 1	0.9	OR	BS1 0.987	BS2 0.950			0.987
49	CONTROLLER 2	0.9	OR	BS1 0.987	BS2 0.950			0.987
50	CONTROLLER 3	0.9	OR	BS1 0.987	BS2 0.950			0.987
51	CONTROLLER LEVEL CROSSBAR NW	0.97	CROSSBAR NW	CONTROLLER 1 0.987	CONTROLLER 2 0.987	CONTROLLER 3 0.987		0.987
52	SERVER 1	0.98	OR	CONTROLLER LEVEL CROSSBAR NW 0.987				0.987
53	SERVER 2	0.98	OR	CONTROLLER LEVEL CROSSBAR NW 0.987				0.987
54	SERVER 3	0.98	OR	CONTROLLER LEVEL CROSSBAR NW 0.987				0.987

Table 3. *Cont.*

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	
				Success Rate S1	Success Rate S2	Success Rate S3	Success Rate S4	
55	SERVER 4	0.98	AND	CONTROLLER LEVEL CROSSBAR NW 0.987				0.967
56	SERVER-LEVEL MESH NW	0.98	MESH NW	SERVER 1 0.967	SERVER 2 0.967	SERVER 3 0.967	SERVER 4 0.967	0.999
57	GATEWAY 1	0.95	OR	SERVER-LEVEL MESH NW 0.999				0.999
58	GATEWAY 2	0.95	OR	SERVER-LEVEL MESH NW 0.999				0.999
59	GATEWAY 3	0.95	OR	SERVER-LEVEL MESH NW 0.999				0.999
60	GATEWAY 4	0.95	OR	SERVER-LEVEL MESH NW 0.999				0.999
61	INTERNET	0.98	OR	GATEWAY 1 0.999	GATEWAY 2 0.999	GATEWAY 3 0.999	GATEWAY 4 0.999	0.999

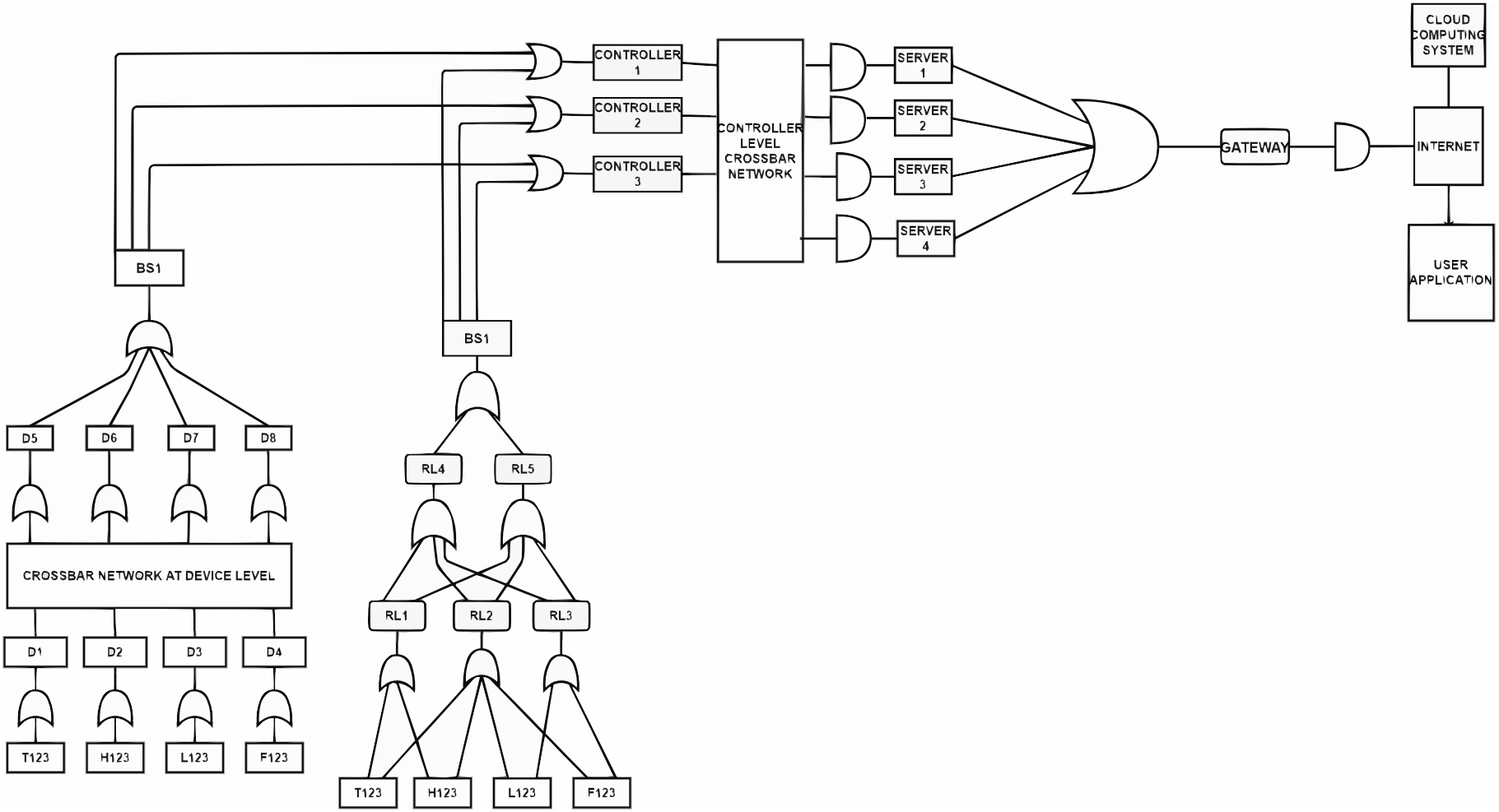

Figure 6. FTA diagram for the example IoT network.

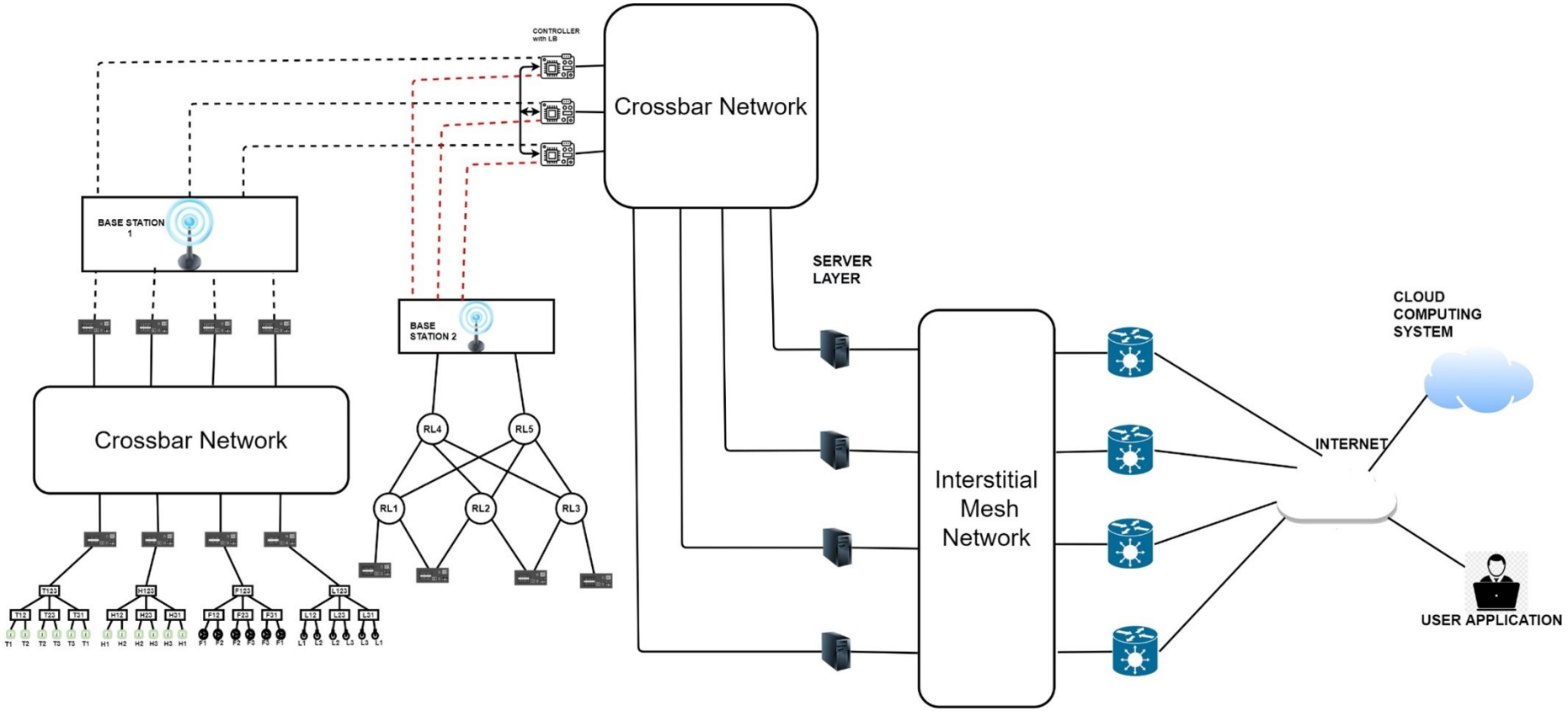

Figure 7. Linearized revised IoT diagram.

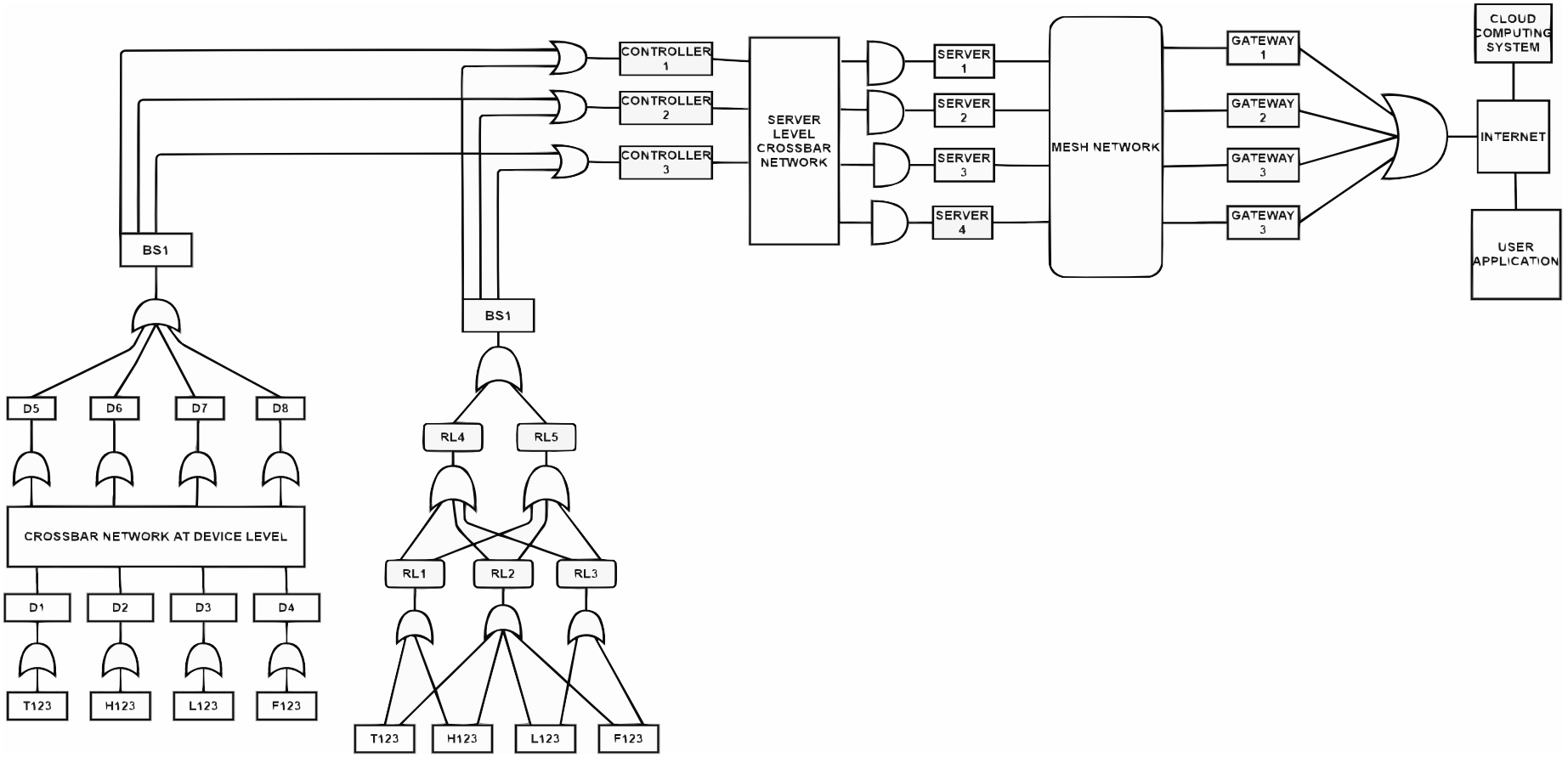

Figure 8. Revised FTA diagram.

8. Discussion

8.1. Stage-Wise Performance Analysis, Including the Changes Made in the Gateway Layer

Rectangular and interstitial mesh networks with built-in selected redundancy will provide very high-level fault tolerance of the IoT network. A probability model derived for the rectangular and interstitial mesh will help to compute the gateway layer fault tolerance. Improving the fault tolerance in the gateway layer will help to achieve very high fault tolerance considering the entire network. The gateway layer of the IoT network is the most crucial, as most computing is done in this layer.

A stage-wise fault tolerance improvement analysis is provided in Table 4. It can be seen from the table that the IoT network's fault tolerance was improved by 11% when a rectangular and interstitial mesh network was introduced in the service layer (FTA changed from 0.980 to 0.9900).

Table 4. Enhancements in different layers—enhancement in fault tolerance of the IoT network.

Serial Number	Type of Network	Fault Tree Value
1	Example IoT network (prototype) [53]	0.717
2	Prototype with modified device layer [54] (implemented fault prediction, mitigation and crossbar network)	0.827
3	The prototype has been modified at the device and base station levels (by introducing dual networks to connect to two base stations and determining the quickest path for communication through the second base station) [55]	0.948
4	Prototype with modifications made at the device and base station levels, with load balancing at the controller layer (controllers interconnected over an I2C network, with middleware implemented within the controllers) and linking the controllers to the service's servers through a crossbar network	0.980
5	Example IoT network with modifications made at the device level and base station levels, with load balancing at the controller layer (controllers interconnected through an I2C network and middleware implemented within the controllers), connecting the controllers to the service's servers through a crossbar network and implementing the prediction model to predict the missing data	0.980
6	Example IoT network with modifications made at the device level and base station level, with load balancing at the controller layer (controllers interconnected through an I2C network and middleware implemented within the controllers), connecting the controllers to the service's servers through a crossbar network and implementing the prediction and estimating the missing data. The service layer also implements a rectangular and interstitial mesh network to link the service's server to the gateways.	0.999

8.2. Failure Analysis of the Proposed Gateway Layer and the Related Comparative Models Based on Node Failures

Failure analysis considers the revised IoT network implemented with rectangular and interstitial mesh networks in the gateway layer and another related model (NVD and HVD). The analysis is presented in Table 5. The number of paths available for communication is reduced concerning NVD and HVD as the failure of the number of nodes increases. The fault tolerance of the mesh network implemented based on NVD and HVD is zero when there is a situation in which four nodes fail simultaneously. The fault tolerance of the system with the interstitial mesh network remains the same, even if four nodes fail, due to the availability of the redundant nodes. Moreover, there is no empirical formulation for the computation of the fault tolerance of the mesh network when NVD or HVD is used.

Table 5. Failure analysis and comparison of rectangular and interstitial mesh with NVD and HVD models in the gateway layer.

Parameter	Mesh with Interstitial Mesh	NVD [21]	HVD [51]
Existence of empirical formulation for computation of FTA	Yes	No	No
Total number of nodes in the network	24	16	16
Number of paths existing in the network	16	16	16
Number of paths available when a node fails	16	12	12
Number of paths available when two nodes fail	16	8	8
Number of paths available when three nodes fail	16	4	4
Number of paths available when four nodes fail	16	0	0
FTA of network when four nodes fail	0.999	0.000	0.00

At best, the NVD and HVD methods can compute the fault tolerance of the gateway layer, whereas the interstitial mesh-based method helps to compute the fault tolerance of the entire network.

9. Conclusions

1. The gateway layer is the most critical in an IoT network, as most of the communication is undertaken in this layer, connecting the internal IoT systems to the external cloud related to the network. Computing is done in the service layer, and the data are moved to the gateway layer for onward transmission to the cloud.
2. The speed at which the servers transmit data to the gateway layer is slow, while the gateway layer transmits the data at very high speeds to the cloud.
3. The equipment (routers and gateways) included in the gateway layer is expensive, and creating excessive redundancy leads to much larger expenses. This cost is not justified when the gateway layer is highly reliable.
4. It is risky if no redundancy is created in this area; entire networks will fail, causing disasters. Therefore, a low-cost solution is required to create redundancy in the gateway layer.
5. Creating a low-cost solution in the gateway layer can be achieved by implementing interstitial mesh networks in the layer, which introduces redundancy to 25%.
6. It has been proven that the IoT network will cater for four node failures using the interstitial mesh network. The failure of four nodes simultaneously is impossible, and the strategy presented in this paper addresses the possibility of the failure of several nodes simultaneously.
7. Implementing an interstitial mesh network in the gateway layer increases the fault tolerance of the IoT network by 11%. An empirical formulation is developed, which helps to compute the fault tolerance of the IoT network when an interstitial mesh network is implemented in the service layer. This is not the case with other models.
8. Interstitial mesh networks in the gateway layer can readily link to the topologies in other network layers.

Author Contributions: S.K.R.J.: writing original draft preparation, writing review and editing, B.C.: Conceptualization, Methodology, Investigation, S.B.J.: Resources, Supervision, Project administration, B.K.D.: Formal Analysis, Data Curation, Visualization, R.B.: Software. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Data Availability Statement: Data is presented in the paper.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Atzori, L.; Iera, A.; Morabito, G. The Internet of Things: A survey. *Comput. Netw.* **2010**, *54*, 2787–2805. [CrossRef]
2. Kawamoto, Y.; Nishiyama, H.; Kato, N.; Yoshimura, N.; Yamamoto, S. Internet of Things (IoT): Present state and prospects. *IEICE Trans. Inf. Syst.* **2014**, *E97.D*, 2568–2575. [CrossRef]
3. Rawat, P.; Singh, K.D.; Bonnin, J.M. Cognitive radio for M2M and Internet of Things: A survey. *Comput. Commun.* **2016**, *94*, 1–29. [CrossRef]
4. Pan, J.; McElhannon, J. Future edge cloud and edge computing for Internet of Things applications. *IEEE Internet Things J.* **2018**, *5*, 439–449. [CrossRef]
5. Adhinugraha, K.; Rahayu, W.; Hara, T.; Taniar, D. On Internet-of-Things (IoT)gateway coverage expansion. *Future Gener. Comput. Syst.* **2020**, *107*, 578–587. [CrossRef]
6. Singh, F.; Kotagi, V.; Siva Ram Murthy, C. Parallel opportunistic routing in IoT networks. In Proceedings of the 2016 IEEE Wireless Communications and Networking Conference, Doha, Qatar, 3–6 April 2016; pp. 1–6. [CrossRef]
7. Zhu, Q.; Wang, R.; Chen, Q.; Liu, Y.; Qin, W. IoT gateway: Bridging wireless sensor networks into Internet of Things. In Proceedings of the 2010 IEEE/IFIP International Conference on Embedded and Ubiquitous Computing, Hong Kong, China, 11–13 December 2010; pp. 347–352. [CrossRef]
8. Sastry, J.K.R.; Sri, Ramya, G.; Niharika1, V.M.; Sowmya, K.V. Performance Optimization of IoT Networks Within the Gateway Layer. *Recent Adv. Comput. Sci. Commun.* **2020**, *13*, 1338–1346. 9. [CrossRef]
9. Liu, Y.; Tong, K.F.; Qiu, X.; Liu, Y.; Ding, X. Wireless mesh networks in IoT networks. In Proceedings of the 2017 International Workshop on Electromagnetics: Applications and Student Innovation Competition, London, UK, 30 May–1 June 2017; pp. 183–185. [CrossRef]
10. Sahni, Y.; Cao, J.; Zhang, S.; Yang, L. Edge mesh: A new paradigm to enable distributed intelligence in the Internet of Things. *IEEE Access* **2017**, *5*, 16441–16458. [CrossRef]
11. Tang, X.; Tan, L.; Hussain, A.; Wang, M. Three-dimensional Voronoi diagram-based self-deployment algorithm in IoT sensor networks. *Ann. Telecommun.* **2019**, *74*, 517–529. [CrossRef]
12. Lin, Q.; Song, H.; Gui, X.; Wang, X.; Su, S. A shortest path routing algorithm for unmanned aerial systems based on grid position. *J. Netw. Comput. Appl.* **2018**, *103*, 215–224. [CrossRef]
13. Liu, X.; Wang, X.; Jia, J.; Huang, M. A distributed deployment algorithm for communication coverage in wireless robotic networks. *J. Netw. Comput. Appl.* **2021**, *180*, 103019. [CrossRef]
14. Zhu, C.; Zheng, C.; Shu, L.; Han, G. A survey on coverage and connectivity issues in wireless sensor networks. *J. Netw. Comput. Appl.* **2012**, *35*, 619–632. [CrossRef]
15. Safar, M.; Ibrahimi, D.; Taniar, D. Voronoi-based reverse nearest neighbour query processing on spatial networks. *Multimed. Syst.* **2019**, *15*, 295–308. [CrossRef]
16. Tran, Q.T.; Taniar, D.; Safar, M. Reverse k nearest neighbour and reverse farthest neighbour search on spatial networks. In *Lecture Notes in Computer Science*; including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics, LNCS; Springer: Berlin/Heidelberg, Germany, 2009; Volume 5740, pp. 353–372.
17. Abeywickrama, T.; Cheema, M.A.; Taniar, D. K-nearest neighbours on road networks: A journey in experimentation and in-memory implementation. *arXiv* **2016**, arXiv:1601.01549. http://doi.org/10.14778/2904121.2904125.
18. Xuan, K.; Zhao, G.; Taniar, D.; Rahayu, W.; Safar, M.; Srinivasan, B. Voronoi-based range and continuous range query processing in mobile databases. *J. Comput. Syst. Sci.* **2011**, *77*, 637–651. [CrossRef]
19. Kolahdouzan, M.; Shahabi, C. Voronoi-based K nearest neighbour search for spatial network databases. In Proceedings of the Thirtieth International Conference on Very Large Data Bases—Volume 30, Toronto, ON, Canada, 31August–3 September 2004; pp. 840–851.
20. Cheema, M.A.; Zhang, W.; Lin, X.; Zhang, Y.; Li, X. Continuous reverse k nearest neighbours queries in Euclidean space and spatial networks. *VLDB J.* **2012**, *21*, 69–95. [CrossRef]
21. Yang, S.; Cheema, M.A.; Lin, X.; Wang, W. Reverse k nearest neighbours query processing. *Proc. VLDB Endow.* **2015**, *8*, 605–616. [CrossRef]
22. Shao, Z.; Taniar, D.; Adhinugraha, K.M. Voronoi-based range-kNN search with map grid in a mobile environment. *Future Gener. Comput Syst.* **2017**, *67*, 305–314. [CrossRef]
23. Lee, I.; Lee, K. Higher order Voronoi diagrams for concept boundaries and tessellations. In Proceedings of the 6th IEEE/ACIS International Conference on Computer and Information Science (ICIS 2007), Melbourne, VIC, Australia, 11–13 July 2007; pp. 513–518. [CrossRef]
24. Gummadi, K.P.; Saroiu, S.; Gribble, S.D. King: Estimating Latency Between Arbitrary Internet end Hosts. In Proceedings of the 2nd Internet Measurement, Workshop (IMW 2002), Marseille, France, 6–8 November 2002; pp. 5–18.
25. Okabe, A.; Boots, B.; Sugihara, K.; Chiu, S.N. *Spatial Tessellations: Concepts and Applications of Voronoi Diagrams*; Wiley Series in Probability and Statistics; Wiley: Hoboken, NJ, USA, 2009.
26. Okabe, A.; Satoh, T.; Furuta, T.; Suzuki, A.; Okano, K. Generalized network Voronoi diagrams: Concepts, computational methods, and applications. *Int. J. Geogr. Inf. Sci.* **2008**, *22*, 965–994. [CrossRef]
27. Saha, T.; Sunitha, R. A framework for Fault detection and prevention in IOT using Smart Gateway. *JCSE Int. J. Comput. Sci. Eng.* **2018**, *6*, 112–122

28. Ma, G.; Yang, Y.; Xuesong, Q.; Gao, Z.; Li, H. Fault-tolerant topology control for heterogeneous wireless sensor networks using the multi-routing tree. In Proceedings of the 2017 IFIP/IEEE Symposium on Integrated Network and Service Management (IM), Pisa, Italy, 8–11 October 2007; pp. 620–623. [CrossRef]
29. Ismail, M.; Zhuang, W. A distributed multi-service resource allocation algorithm in heterogeneous wireless access medium. *IEEE J. Sel. Areas Commun.* **2012**, *30*, 425–432. [CrossRef]
30. Li, Y.; Yu, J.; Tao, D. Genetic algorithm for spanning tree construction in P2P distributed interactive applications. *Neurocomputing* **2014**, *140*, 185–192. [CrossRef]
31. Karthikeya, S.A.; Vijeth, J.K.; Murthy, C.S.R. Leveraging Solution-Specific Gateways for cost-effective and fault-tolerant IoT networking. In Proceedings of the 2016 IEEE Wireless Communications and Networking Conference, Doha, Qatar, 15 September 2016. [CrossRef]
32. Kim, M.; Ko, I. An efficient resource allocation approach based on a genetic algorithm for composite services in IoT environments. In Proceedings of the 2015 IEEE International Conference on Web Services, New York, NY, USA, 27 June–2 July 2015; pp. 543–550. [CrossRef]
33. Takahashi, R.; Ota, M.; Fukazawa, Y. Fault-Tolerant Topology Determination for IoT Network. In Proceedings of the 13th International Conference on Ubiquitous Information Management and Communication (IMCOM) 2019, January 4–6, 2019, Phuket, Thailand, 2019; pp. 62–76. [CrossRef]
34. Leonardi, L.; Patti, G.; Lo Bello, L. Multi-hop real-time communications over Bluetooth low-energy industrial wireless mesh networks. *IEEE Access* **2018**, *6*, 26505–26519. [CrossRef]
35. Rondon, R.; Mahmood, A.; Grimaldi, S.; Gidlund, M. Understanding the performance of Bluetooth mesh: Reliability, delay, and scalability analysis. *IEEE Internet Things J.* **2019**, *7*, 2089–2101. [CrossRef]
36. He, B.; Xie, B.; Agrawal, D.P. Optimizing the internet gateway deployment in a wireless mesh network. In Proceedings of the 2007 IEEE International Conference on Mobile Adhoc and Sensor Systems, Pisa, Italy, 8–11 October 2007. [CrossRef]
37. Seyedzadegan, M.; Othman, M.; Ali, B.M.; Subramaniam, S. Zero-degree algorithm for internet gateway placement in backbone wireless mesh networks. *J. Netw. Comput. Appl.* **2013**, *36*, 1705–1723. [CrossRef]
38. Shih, C.-S.; Wu, G.-F. Multiple protocol transport network gateway for IoT systems. *ACM SIGAPP Appl. Comput. Rev.* **2016**, *15*, 7–18. [CrossRef]
39. Jean-Philippe, V.; Mario, P.; Piet, D. Network recovery: Protection and restoration of optical, SONET-SDH, IP, and MPLS. In *Morgan Kaufmann, Amsterdam, Series*, 1st ed.; Elsevier: Amsterdam, The Netherlands, 2004. [CrossRef]
40. Médard, M.; Barry, R.A.; Finn, S.G.; He, W.; Lumetta, S.S. Generalized loop back recovery in optical mesh networks. *IEEE/ACM Trans. Netw.* **2002**, *10*, 153–164. [CrossRef]
41. Choi, H.; Subramaniam, S.; Choi, H.A. Loopback recovery from double-link failures in optical mesh networks. *IEEE/ACM Trans. Netw.* **2004**, *12*, 1119–1130. [CrossRef]
42. Lakshmanan, S.; Sivakumar, R.; Sundaresan, K. Multi-gateway association in wireless mesh networks. *Ad Hoc Netw.* **2009**, *7*, 622–637. [CrossRef]
43. Kawai, T.; Yusa, N.; Mineno, H. Implementation and Evaluation of adaptive multi-gateway mesh network. In Proceedings of the 2014 International Conference on Network-Based Information Systems, Salerno, Italy, 12–14 September 2014; pp. 61–68. [CrossRef]
44. Adhinugraha, K.; Rahayu, W.; Hara, T.; Taniar, D. Backup gateways for IoT mesh network using order-k hop Voronoi diagram. *World Wide Web* **2021**, *24*, 955–970. [CrossRef]
45. Kim, K.-H.; Shin, K.G. Self-reconfigurable wireless mesh networks. *IEEE/ACM Trans. Netw.* **2011**, *19*, 393–404. [CrossRef]
46. Gilani, S.S.A.; Qayyum, A.; Rais, R.N.B.; Bano, M. SDNMesh: An SDN-based routing architecture for wireless mesh networks. *IEEE Access* **2020**, *8*, 136769–136781. [CrossRef]
47. Sarkar, S.; Yen, H.H.; Dixit, S.; Mukherjee, B. RADAR: Risk-and-delay aware routing algorithm in a hybrid wireless-optical broadband access network, WOBAN. In Proceedings of the Optics InfoBase Conference Papers, Washington, DC, USA, 25–29 March 2007; pp. 4–6.
48. Mahiddin, N.A.; Sarkar, N. An efficient gateway routing scheme for disaster recovery scenarios. In Proceedings of the International Conference on Information Networking, Kuala Lumpur, Malaysia, 9–11 January 2019; pp. 204–209. [CrossRef]
49. Minh, Q.T.; Nguyen, K.; Borcea, C.; Yamada, S. On-the-fly establishment of multi-hop wireless access networks for disaster recovery. *IEEE Commun. Mag.* **2014**, *52*, 60–66. [CrossRef]
50. Kiki, A.; Wenny, R.; Takahiro, H.; David, T. Measuring fault tolerance in IoT mesh networks using Voronoi diagram. *J. Netw. Comput. Appl.* **2022**, *199*, 103297. [CrossRef]
51. Kiki, A.; Wenny, R.; Takahiro, H.; David, T. On Internet-of-Things (IoT) gateway coverage expansion. *Future Gener. Comput. Syst.* **2020**, *107*, 578–587
52. Moghaddam, M.T.; Muccini, H. Fault-Tolerant IoT. In *Software Engineering for Resilient Systems. SERENE 2019. Lecture Notes in Computer Science*; Calinescu, R., Di Giandomenico, F., Eds.; Springer: Cham, Switzerland, 2019; Volume 11732, pp. 67–84. [CrossRef]
53. Bhupati, C.; Sastry, K.R.J. Hybrid models for computing fault tolerance of IoT networks. *TELKOMNIKA (Telecommun. Comput. Electron. Control.)* **2023**, *21*, 333–345.

54. Bhupati, R.R.; Sastry, J.K.R. Improving the fault tolerance of an IoT, Network through power detection and Isolation. *J. Intell. Fuzzy Syst.* 2023, *submitted*.
55. Bhupati, J.K.R.; Ch, B.; Budaraju, R. Implementing dual base stations within an IoT network to sustain an IoT network's fault tolerance through an Efficient Path Finding Algorithm. *Sensors* **2023**, *23*, 4032.