A Study on using Machine Learning Algorithms to Improve IoT Network Reliability

Kevin Ramirez
Computer Engineering & Computer Science
CSU Long Beach
Long Beach, United States of America
Kevin.Ramirez03@student.csulb.edu

***Abstract* - An Internet of Things (IoT) Network is a network of devices connected through the internet to send and receive data to one another. These networks are often developed to take in information from sensors and send said data to another device for analysis and display. Past works have demonstrated the use of Machine Learning as a Service (MLaaS) fused with IoT to enhance the quality of the data obtained from sensory devices as well as using MLaaS to improve the functionality of various IoT applications but do not account for the sensory device's operation itself. To counter this, introducing the use of machine learning algorithms may improve the reliability of these networks during a failure in one or more devices. These predictive algorithms are not replacements of the sensory devices, rather come into use when one or more devices fail and are aimed to keep the IoT network operational for a limited number of device failures. This paper will study previous works attempts to enhance IoT networks reliability as well as simulate a virtual IoT network with device failures using Node-Red API. The expected result will be an increase in the IoT network's reliability as faults which occur over time will be masked with near approximations of the live reading while minimizing hardware redundancy.**

Key Words – Machine Learning, Internet of Things, Network, Fault Tolerance, Topology, Prediction, Architecture, Application, Sensor, Device, Reliability

I. INTRODUCTION

The Internet of Things (IoT) is a distributed computing paradigm which connects multiple devices to the internet and permits message passing between one another. Often, devices within an IoT network can obtain information from a sensor and send the information through the internet to another device or external service for processing. This paradigm has a wide range of applications, ranging from turning on the light in your room when a smartphone sends a message to a device controlling the light bulb [5], to a vast network of physical sensors obtaining weather information and relaying the data over the internet to a processing service which determines whether a natural disaster will occur [2]. The network's ability to use numerous devices to obtain sensory data makes it a target for machine learning, as this property assist greatly in training machine learning models using the large data sets.

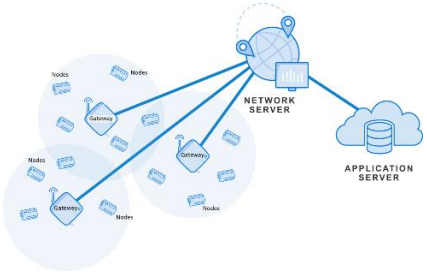

Figure 1: Diagram of General IoT Network

However, the network is only useful if the network has one or more devices operating, connected to the internet, and sending data for processing. This raises a question on how to improve the reliability of these networks and prevent faulty network devices degrading the network's operation. Many studies have examined several different methods to boost the reliability of these IoT networks, however, require the use of additional hardware to replace fault devices or provide a solution which only routes messages around faulty devices.

This project will provide a software solution which utilizes machine learning's capabilities to temporarily replace a faulty device by obtaining approximations of the current data and provide an approximation of the live reading. The project will be applicable to specific subsets of IoT networks which devices neighbor one another by proximity, have large data sets to train a machine learning model, and can detect which device in the network is inoperable or disconnected. Ideally, the project will provide the current readings of live information despite the disconnection of live devices without the use of additional hardware.

II. MACHINE LEARNING AND INTERNET OF THINGS TECHNOLOGY

One of the most important fields of study in recent times is the field of machine learning, which utilizes a teaching and learning approach to train a machine in understanding data when provided. When executed properly, this technology can automate many processes which require some form of classification, provide innumerable amounts of data much too large to manually processes, and difficult to program the machine how to accomplish certain tasks. Machine learning comes in various forms: supervised, unsupervised, semi-guided, and reinforcement learning.

Depending on the application as well as the nature of the data, systems will utilize one of these four forms of machine learning training.

- **Supervised:** Training a machine learning model with labelled data and a known desired outcome.
- **Unsupervised:** Training a machine learning model with unlabeled data and an unknown outcome.
- **Semi-Supervised:** A combination of supervised and unsupervised, whereas only a portion of the data is known and labelled in advanced.
- **Reinforcement Learning:** A training method which uses its previous correct and incorrect assessments to enhance future outcomes.

To utilize the benefits afforded by machine learning, understanding the various models machine learning services provides as well as understanding the nature of the problem which needs to be solved can result in a successful machine capable of high accuracy rates in accomplishing the task it is trained to do. In the article *Location Prediction on Twitter using Machine Learning Techniques* [3], the project extracted thousands of tweets across various accounts and organized the information to contain only the relevant information: the tweet, the user's home location, and the general region in which the tweet was made. The project also understood the task was to predict the user's location based on such given information, understanding in advance what the desired outcomes were and fell in the category of supervised learning. Afterwards, it selected and trained three machine learning models with the dataset and resulted in three models with various accuracy ratings when predicting the location of a user based on their tweet: Naïve bayes at 43.67% accuracy, SVM with 86.78% accuracy, and Decision Tree at 99.6% accuracy.

With machine learning's need for large training data set, its capabilities in making assessments on provided data and IoT's ability to produce large quantities of data, a natural cross of these two technologies have emerged. In the article *A Literature Study on Machine Learning Fusion with IOT* [4], a multitude of applications have been explored which exploits the benefits provided by machine learning technology trained with the data generated by IoT networks. In one example, an IoT network comprised of sensors which monitored patients' health was used to train an unsupervised machine learning model to detect anomalies in a patient's health, reducing human error in diagnosis and raising early detection rates of diseases [4]. Another example demonstrated how a supervised machine learning model trained with five years of traffic data across three cities in London helped reduce traffic congestion through automated traffic control [4].

III. SPECIFIC INTERNET OF THINGS NETWORK PROPERTIES

Enhancing the fault tolerance of an internet of things network is a difficult task without any single solution, as answering this question will vary between applications. Specifically, enhancing the fault tolerance in any internet of things network will vary due to the variety in network topology, what the application is attempting to accomplish, how the devices are set up, what the devices are capable of, and any additional technologies or services used in the IoT network. For the purposes of this paper, the project will focus on IoT network which contain the following properties:

- Devices obtain real-time information through sensors.
- Devices will neighbor one another.
- Devices are capable of sharing information.
- Devices, through one or more connections, send sensor data to an external service for aggregation and processing.
- Devices and the external processing services are always online in unison.
- The network consists of numerous data generating devices.
- The network is capable of accumulating large data.

These limitations in what the network must contain are useful in defining a subset of IoT networks to which the paper studies, as networks which do not contain one or more of the defined limitations will not benefit from the observations and solutions provided in this paper. Furthermore, it should be noted that the proposed solution provided in this paper applies primarily to IoT networks which consists of these properties.

IV. RELATED WORKS

Research conducted in the past examined various methods to enhance the fault tolerance of IoT networks. However, these studies often bring into question whether such a solution is effective across many IoT networks with the properties described previously. They may prove as a method which may greatly increase the fault tolerance for certain IoT applications but come at an additional cost which may hinder the performance of the overall network if not met.

A. Use of Additional Hardware

The use of additional hardware is an intuitive and common solution which answers the question of what to do when a device fails. In the article *Towards a New Approach to IoT Fault Tolerance* [5] , an example IoT network comprised of a motion detector, a light bulb, and two processing hubs connected to the internet (Figure 2) demonstrated how any connection or device faults in the network will render the network inoperable. If any device

fails to receive or send data from the motion detector to the lightbulb, then the IoT network will remain inoperable unless devices are repaired or replaced.

Figure 2: IoT Network to Toggle a Light Bulb using Motion Sensor

The most common and intuitive solution to small scale networks such as in Figure 2 may be to use a hardware redundancy technique known as triple modular redundancy, which triples the hardware in the network and uses an additional voting system to vote on which output is correct. This allows the network to circumvent faulty devices and keep the network operational. In the given example, tripling each hardware component would allow the network to continually send and receive motion information despite failures at any stage of the network (Figure 3). Should a single motion sensor fail in any capacity, the extra motion sensors would remain operational and provide the data to each available processing hub.

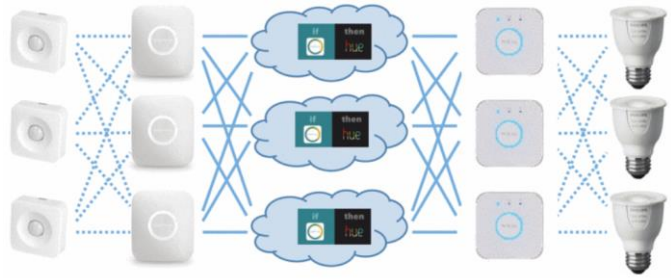

Figure 3: Use Case of Triple Modular Redundancy

Although using additional hardware to mask faults within an IoT network is effective, it is not a viable solution for large scale networks with the described set of properties [5]. The largest issue with this solution is increasing the monetary cost of the network by up to triple the original cost [5]. Of course, variations on the configuration may bring the cost down, but will still demand extra hardware in some capacity to ensure faults do not inhibit the network's operation, thus demanding a greater cost than the minimum required to operate the network.

B. *Use of Routing Software*

Other works have proposed the use of a software solution to work around faults which may occur during the network's operation. An article titled *Fault-Tolerant Techniques for the Internet of Military Things* [1] proposes the use of an experimental method which creates clusters of devices (Figure 4) in the device layer of the IoT network to permit messages from other sensory devices to work around faulty devices. The clustering of devices will permit a coordinator device to detect which devices within its cluster is currently failing, then re-routes messages from other currently functional devices around the faulty device towards the network gateway, to which information is then passed onto an external processing service or other clusters [1].

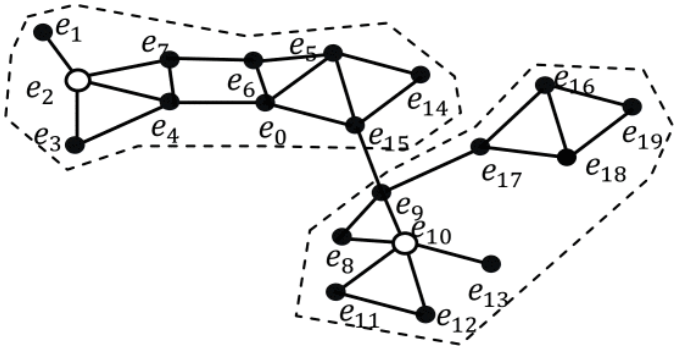

Figure 4: Clustering of Devices within a Network

This solution, if implemented correctly, may greatly enhance the fault tolerance of an IoT network with the provided properties. Devices which are inoperable are avoided, allowing the network to only account for viable data provided by live and functional devices. However, faults within these clusters may continue to grow and may cause disconnections between a cluster and the gateway to the internet, causing large number of devices to lose connection to the overall network. A fault of this scope in size may greatly reduce the functionality of the network, as well as demand immediate attention by the end user to replace or repair faulty devices.

C. *Revised Internet of Things Architecture using FOG Computing*

One solution to the question of improving fault tolerance is the use of an extra computing paradigm, known as FOG computing, to layer the network into three sections with the fog layer in between. Fog computing is a computing architecture which acts as a rest stop for data to accumulate and process, then send processed and more condensed data towards a data aggregation server such as the cloud. Due to its design, applications which utilize fog computing often achieve lower latency and an increase in computing power, relieving some of the stress from the application server [2]. In the article *How to Improve Fault Tolerance in Disaster Predictions: A Case Study about Flash Floods Using IoT, ML and Real Data* [2], the work discussed an IoT network (Figure 5) capable of receiving sensory information across a large network of devices, then utilize a fog layer in between the devices and cloud to accumulate and process smaller subsets of data before sending it to the cloud. In addition, the devices in tier 1 of this IoT network can share information between one another as well, and devices of tier 1 and tier 2 are close in proximity to one another [2].

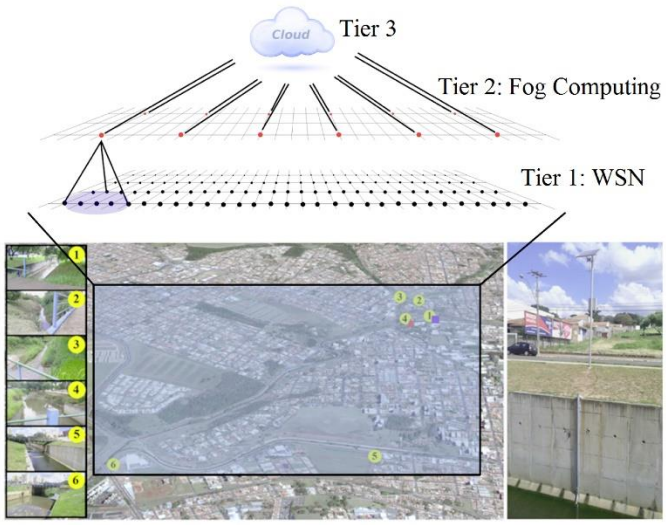

Figure 5: IoT Network utilizing FOG Computing Layer

When faults in this architecture occurs, devices from a lower tier will temporarily assume the role of the faulty device [2]. For example, if a device in tier 2 fails, a tier 1 device which previously sent data to the faulty tier 2 device will be elected as a temporary replacement, assuming all processing and data accumulation roles of the previous tier 2 device. However, this architecture may encounter issues discussed in both Section 4-A and Section 4-B. The use of an additional computing paradigm will inevitably increase costs of operation, as using an extra computing layer will behave similarly to adding extra hardware to the architecture. Alongside this, the paradigm behaves similarly to the network clusters discussed in Section 4-B, as devices in lower tiers send sensor data to a pre-defined tier 2 device. Should a tier 2 device fail, data obtained by the tier 1 devices under the faulty tier 2 device may not be sent towards the cloud service, as this architecture opts for a local solution when this occurs instead, resulting in loss of sensor devices like clusters in Section 4-B.

V. USING MACHINE LEARNING TO ENHANCE INTERNET OF THINGS FAULT TOLERANCE

Understanding how previous attempt to enhance the fault tolerance of IoT networks with the properties listed in Section 4, revisiting machine learning and its applications in IoT networks brings forth the following question, can machine learning be applied to mask device faults in IoT networks? Much like hardware redundancy, the objective of this theory would be to use machine learning to mask device failures when they occur temporarily until the device is repaired or replaced. Using the historical data provided by the by the IoT network, a machine learning model will be trained to understand what readings are to be expected, then when provided with data by the faulty device's neighbors should it be possible to predict with high accuracy the current reading. In this circumstance, topology plays a much larger role in how useful the data provided by the model is. In Figure 6, the devices in the IoT network are represented as nodes, whereas neighbors between one another are represented as edges. Each device is also connected to the internet individually as well to send generated data to an application server. Devices which encounter a failure in any capacity are dotted, and therefore will be temporarily replaced by a machine learning model which will continue providing approximate values in place of the faulty device. This topology could, in theory, permit the network to encounter up to an estimated (n/2+1)/(n) percentage of device failures and remain operational, whereas n equals to the number of data generating devices in the network.

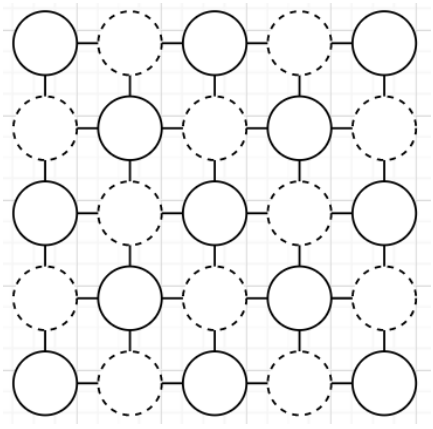

Figure 6: 5x5 Grid Topology with Intermittent Faulty Devices

However, IoT networks with certain topologies will not benefit from the predictive processes machine learning provides, as the methodology requires devices to have a unique subset of neighbors. Consider Figure 7, in which several nodes share a singular neighbor. As devices which neighbor the central node encounter faults, the machine learning model will rely on information provided by the faulty device's neighbor to make approximations. However, due to these nodes sharing a singular neighbor, the node will provide the same information across all faulty devices, rendering the use of machine learning ineffective and data provided redundant. Alongside this, if the central device which neighbors several devices encounter an issue, the remaining nodes will no longer have a neighbor to receive approximate data form, disabling the use of the machine learning predictor. As such, topology of the devices within the IoT network plays a crucial role in ensuring devices share and receive unique data.

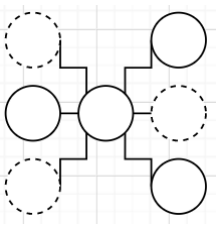

Figure 7: Node Topology which Several Nodes Share a Single Neighbor

VI. SIMULATION

This project is an attempt to recreate the IoT network solution proposed within this paper to determine if utilizing machine learning to mask device level faults is feasible. As such, certain conditions must be met:

1. The simulation must replicate a topology which permits nodes to have a unique subset of neighbors. For this condition, I replicated the 5x5 grid topology in Figure 6.
2. The simulation must permit each node to generate data, acting as some form of data gathering sensor. For this condition, each node will obtain a set of random weather conditions and classify if the weather is cold, moderate, or hot.
3. The simulation must permit each node to send and store information at an application server external to the devices. For this condition, nodes are scripted to send and update documents within an external database outside the simulation.
4. The simulation must utilize a machine learning model to utilize data generated by the nodes for training and testing of accuracy. For this condition, I utilized several machine learning libraries from python: Scikit-Learn, Numpy, and Pandas.
5. The simulation must provide a reasonable accuracy score given the application. This will vary between each use case, so for my simulation I will attempt to achieve 95% accuracy or greater to consider it a success.

In my example simulation I utilized an API known as Node-Red to configure nodes which carry out individual tasks to create visual representations of the network instead of writing code. The project will simulate a 5x5 grid of weather gathering sensors (Figure 6) which will generate the following information: season, cloud density, wind speeds, humidity, and climate estimation. Information of each sensor will be stored as JSON objects (Figure 8) inside a local database service, MongoDB, which will continually receive updates from the simulated sensor devices in Node-Red.


```
State: "California"
City: "Los Angeles"
ID: 0
X: 0
Y: 0
Date: Array
Temperature: Array
Cloudy: Array
Season: Array
Humidity: Array
WindSpeed: Array
Neighbors: Array
    0: 5
    1: 1
```


Figure 8: Example of Data Stored in External Application Database

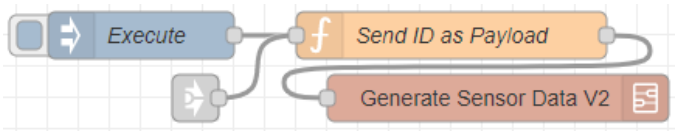

Figure 9: Node-Red Configuration of Simulated Sensory Device

The simulated devices (Figure 9) are configured to execute once every second until the simulation ends. At each execution weather data is generated, classifying the current weather as cold, moderate, or hot, then updating the data in the external database with the generated information (Figure 10). In this application, a defined set of rules determines the classification of the weather, which in turn ensures the machine learning model will undergo supervised training. This is to verify afterwards whether the machine learning model has or has not correctly made a prediction.


```
State: "California"
City: "Los Angeles"
ID: 0
X: 0
Y: 0
Date: Array
    0: 2021-08-16T01:35:05.419+00:00
Temperature: Array
    0: "COLD"
Cloudy: Array
    0: 8.73
Season: Array
    0: 1
Humidity: Array
    0: -1.1999999999999997
WindSpeed: Array
    0: 6
Neighbors: Array
    0: 5
    1: 1
```


Figure 10: Example of a Single Data Entry Generated by Sensor with An ID Zero

Once a sufficient size of data is generated across the 5x5 grid of sensors, pre-processing the data to only contain information relevant to the machine learning model occurs (Figure 11). In this simulation, the machine learning model used is the decision tree classifier model, which works well as the generated data are a set of conditions which determine the climate.

	A	B	C	D	E
1	Season	Cloudy	Humidity	Wind Speed	Climate
2	1	8.73	-1.2	6	COLD
3	2	0.27	-1.68	5	MODERATE
4	4	0.99	-1.92	1	COLD
5	1	4.77	-4.8	5	COLD
6	3	0.18	-5.16	2	COLD
7	3	0.81	3.36	1	MODERATE
8	1	1.62	-2.28	3	COLD
9	1	5.94	-0.6	2	COLD
10	1	3.96	-5.16	6	COLD

Figure 11: Example of Pre-Processed Data for Machine Learning Use

The data, partitioned into a training and testing set, is then fed to train the machine learning model (Figure 12).

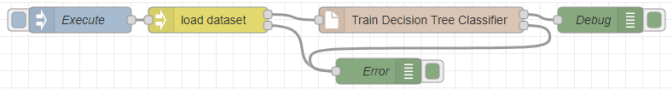

Figure 12: Node-Red Configuration which trains a Decision Tree Classifier

Once the model is trained, it can now be tested with the unseen test data to determine its accuracy in predicting the climate (Figure 13).

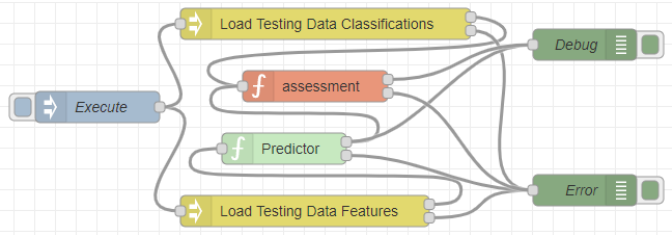

Figure 13: Node-Red Configuration which assesses Decision Tree Classifier Accuracy

In this execution instance, the decision tree model was trained with a dataset of 3,060 data points and tested with 765 unseen data. Assessing the accuracy of the model after training resulted in an accuracy rate of 96.07%. Of course, the accuracy of the model is subject to vary when using different models, tweaking the parameters of the model, or utilizing different datasets, all depending on the application of the IoT network and configuration of the machine learning model. Finally, the model can be used to begin making predictions on what the current climate is given data points by the faulty device's neighbors (Figure 14).

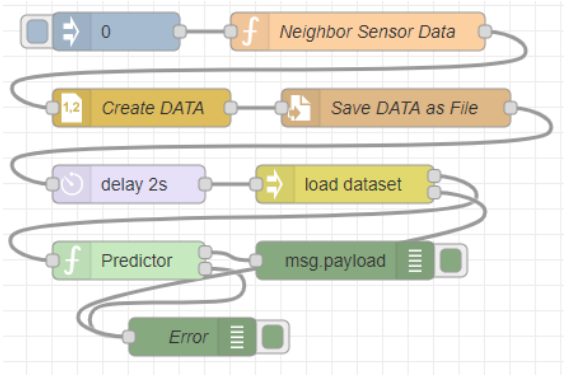

Figure 14: Node-Red Configuration which obtains Data provided by Neighbors to Predict Climate at the Faulty Device.

Once a model is trained and assessed, it may begin predicting the current climate given the faulty device's neighbor(s) sensor information. With a proper sensor topology, machine learning model, and a sizeable dataset, machine learning may act as a virtual automatic replacement to faulty devices within the IoT network which provide a near 100% accurate reading.

VII. CONCLUSION

The various techniques used in the industry to enhance the IoT networks often resulted in viable but expensive methods which utilize extra hardware or methods which may accumulate numerous device failures. As an alternative, the use of machine learning may improve the reliability of certain IoT networks without implementing hardware redundancy, as IoT networks can gather enough information to train a machine learning model quickly. The simulation, albeit not perfect, is a proof of concept which demonstrates how accurate a well-train model could be, and how it may act as a temporary replacement to a faulty sensor. These models may be trained continually and use the information of active sensors neighboring a faulty one to make a prediction.

ACKNOWLEDGEMENTS

I would like to thank all my friends and family who encouraged me to push myself beyond my limits and assisted in opening free time to allow me to research this paper. I would like to thank all references used within this paper which greatly enhanced my understanding in the topics presented.

REFERENCES

[1]. Chudzikiewicz, Jan, et al. "Fault-Tolerant Techniques for the Internet of Military Things." *IEEEXplore*, IEEE, 21 Jan. 2016, ieeexplore-ieee-org.csulb.idm.oclc.org/document/7389104/authors#authors

[2]. Furquim, Gustavo, et al. "How to IMPROVE Fault Tolerance in Disaster Predictions: A Case Study about Flash Floods Using IOT, ML and Real Data." MDPI, Multidisciplinary Digital

Publishing Institute, 19 Mar. 2018, www.mdpi.com/1424-8220/18/3/907/htm.

[3]. K, Indira, et al. “Location Prediction on Twitter Using Machine Learning Techniques.” *IEEEXplore*, IEEE, 23 Apr. 2019, ieeexplore-ieee-org.csulb.idm.oclc.org/stamp/stamp.jsp?tp=&arnumber=8862768&tag=1.

[4]. Sharma, Kirti, and Rainu Nandal. “A Literature Study on Machine Learning Fusion With IOT.” *IEEEXplore*, IEEE, 10 Oct. 2019, ieeexplore-ieee-org.csulb.idm.oclc.org/document/8862656.

[5]. Terry, Doug. “Toward a New Approach to IoT Fault Tolerance.” *IEEEXplore*, IEEE, 15 Aug. 2016, ieeexplore-ieee-org.csulb.idm.oclc.org/document/7543452/authors#authors.