Fault Tolerance in IoT: Techniques and Comparative Study

Abhay Agrawal
Department of Computer Science and Information Systems
Birla Institute of Technology & Science,Pilani Campus
h20190057@pilani.bits-pilani.ac.in

Devendra Toshniwal
Department of Computer Science and Information Systems
Birla Institute of Technology & Science,Pilani Campus
h20190059@pilani.bits-pilani.ac.in

***Abstract*— Fault tolerance increases system availability and reliability by making systems robust to failures and proactive enough to tackle failures. Fault tolerance can be introduced at different architectural layers of the Internet of Things (IoT), this is because a fault can occur at any of the layers. As for example, motion sensors, and motors can fail at the root layer, network connectivity could be disrupted in network layer, computation and storage nodes can perform erroneously in their layers, so it becomes crucial to introduce fault tolerance in IoT systems at every layer. The study paves the way for classifying current and possible fault tolerant approaches by presenting different techniques(replication, network control etc.), architectural patterns(centralized, hybrid etc.), layers(network, sense etc.) & styles(Microservices, Publish-Subscribe etc.) that can help in making a system fault tolerant efficiently. Paper also discusses current trends in fault tolerance, areas that have been widely worked upon and areas that can act as a future scope, in making IoT based systems fault tolerant and efficient.**

Keywords—Fault Tolerance, Internet of Things, Replication, Reliability, Availability.

I. INTRODUCTION

In order to deliver smart services, IoT is the internal/external communication of intelligent elements [1] through the internet. Reliable and fault-free facilities should be offered by a dependable IoT scheme. A fault is a flaw that impacts the correct functionality within the hardware or software systems [2]. As IoT devices are heterogeneous, highly distributed, battery-powered, and reliant on wireless communication and affected by scalability, it is especially difficult to create a pattern for Fault Tolerance in IoT. The IoT devices that are distributed [3] in nature may cause the system to suffer from server crashes, server omissions, incorrect responses, and arbitrary errors. The reliance on wireless and battery makes the IoT devices hardly recoverable [4]. In addition, being exposed to new equipment and facilities influences the performance of the system.

Although the IoT was launched more than a decade ago [5], its various aspects and quality of services (QoS) such as Fault Tolerance are still being attempted by the researchers to define them well. Therefore the purpose of this research is to define and classify the state of the art of the domain and to highlight the approaches, techniques and architectures that are potentially relevant for modelling IoT with fault tolerance. A comprehensive mapping analysis has been carried out in order to achieve this objective. Based on precise inclusion and exclusion criteria and a detailed review, the primary studies were selected.

The paper is organized in the following sections: In section II, related works in the field of fault tolerance in IoT are discussed. Section III explains the taxonomy, on the basis of which fault tolerance in different systems is compared. The comparison is done in section IV. Section V, presents the current trends in the field of fault tolerance in IoT. Section V concludes the paper.

II. RELATED WORKS

Moghaddam et al. [6] discussed different ways of achieving fault tolerance in IoT systems, fault tolerance aspects and subdomains in fault tolerance. The paper also shows changing and emerging trends in the field of fault tolerance in IoT the study is performed in a systematic mapping way. And paving a foundation for future studies in fault tolerance in Iot domain.

Rullo et al. [7] reviewed fault tolerance techniques based on redundancy that targets availability and data integrity. The paper discusses fault tolerance implementation techniques and approaches at sensing & network layer. The paper reviews recent proposed approaches for achieving fault tolerance, shows how they can be implemented to introduce fault tolerance at device level, overcoming disadvantages of old algorithms.

III. TAXONOMY

The aim of this study is based on the Goal-Question-Metric insights which are as follows:

Purpose: to have a thorough understanding of IoT fault-tolerant systems.

Issue: through the detection, classification and analysis of different approaches, techniques and architectures.

Object: Approaches based on existing IoT frameworks.

Viewpoint: From the perspectives of both research and industry.

We considered all the selected studies afterwards and filtered them according to a set of well-defined criteria for inclusion and exclusion. According to the guidelines, two key drivers have driven the concept of inclusion/exclusion criteria: (i) keeping the focus of the selected papers on the scope of the study; and (ii) avoiding grey or non-scientific work.

A. Architecture Layers

1) Actuator:

Actuators transform an electrical signal into a physical quantity that correlates, such as motion, force, sound, etc. In paper [1], fault tolerance is introduced at actuate layers by making use of multiple devices achieving a common task. In the paper, in order to detect presence of person devices like CCTV, Bluetooth, Wi-Fi, noise detector, are being used.

2) Sensor:

A sensor is a tool capable of detecting modifications in an environment. A sensor is useless on its own but it plays a key role when we use it in an electronic device. A sensor can measure and convert a physical phenomenon (such as temperature, pressure, and so on) into an electrical signal. In paper [4], fault tolerance is introduced at the sense layer. The paper proposes a novel way of detecting fault in sensors by observing the sensor's voltage value, when power is fluctuated. The paper claims that proposed techniques are 99% efficient in detecting the faulty sensor.

3) Processing and Storage:

The output level depends on how often the components of processing and storage are decentralized forced to the edge.

Processing is the execution step for a particular system that can be judged based on the time. Storage is another aspect that can play an important role in effectively storing large volumes of data. In Paper [8] fault tolerance is achieved in edge computing systems by introducing Docker, Apache Kafka and Kubernetes.

4) Network:

In IoT the reliability of networks is also an important aspect to study, network topologies should be simple and adaptable to the changes. Paper [9] focuses on introducing fault tolerance at network layer, in which, a routing algorithm is proposed that searches disjoint routes for message exchanges in the system, making it robust to failure

B. Architectural Patterns

1) Distributed Collaborative:

The pattern of the architecture can be distributed which in turn divides the network and the data into different sites. This can have some advantages and disadvantages as well as described in paper [10] of our study.

2) Centralized:

A centralized architecture means a single or a few organizations are available that have control over the entire network. Note that a centralized approach [11] usually implies one-hop communication for all members of the network, but is typically realized by a multi-hop network in the context of short-range embedded systems.

3) Hybrid

This type of architecture combines both the techniques i.e. centralized and decentralized or distributed. This can result in more improve in the overall performance of the system as a whole as discussed in paper [5].

C. Architectural Styles

1) Microservices:

In IoT systems, microservices and SOA have the same purpose, which is to create one or several applications from a collection of different services. A microservice is a lightweight, single-responsibility program that can be independently deployed, scaled and evaluated. In paper [4] a system is proposed in which microservices run at containers, the system keeps track of the status of each microservice, in case of any failure, first a repairing attempt is made, in case attempts fail a replica is run.

2) Service oriented Architecture (SOA):

Service Oriented Architecture (SOA) put the service at the core of the design of their IoT application. The core application component, in reality, makes the service accessible over a network for other IoT components. Paper [5] presents a Platform-as-a-service (PaaS) to developers, where he can develop IoT based applications using API, modules, frameworks etc.

3) Publish-Subscribe:

Publish/Subscribe is a pattern of messaging aimed at decoupling the sending (publisher) and receiving (subscriber) groups. In paper **[8**] fault tolerance is handled by using Apache Kafka, publish/subscribe style for achieving data replication, showing high performance.

D. Fault-Tolerance Techniques

1) Replication:

Replication is primarily used in the distributed systems research field to provide fault tolerance. In active replication each client request is processed by all the servers. In passive replication there is only one server (called primary) that processes client requests. In paper [12], fault tolerance is achieved by dividing end nodes into different groups, within each group all other nodes act as a backup node for each node.

2) Network Control:

The IoT network is normally split into separate clusters within the network control scheme. A chosen cluster head (CH) makes roll call requests to the other nodes regularly and the failure will be verified if it does not receive a response message. The CH itself does however, establish a single point of failure.

3) Distributed Recovery Block:

In this process, a single program is executed simultaneously on a pair of nodes, one of which is active and the other is inactive. The main active) node performs the task in a no-fault situation and the other node performs the same task in the shadow. Afterwards all results will be checked and the results associated with the main node will be transmitted as the output if the test is passed properly. The shadow node becomes active and generates the outputs if the primary node test fails. If the primary node test fails, the shadow node becomes active and produces the outputs. In paper [3], a system is proposed in which under no-fault condition, main sensors collect and send the data to the central server, but parallel same data is being sensed by shadow (backup) sensor, in case of fault, shadow sensor replaces main sensor.

4) *Time Redundancy:*

At all instruction and task stages, time replication may be done. The software is duplicated at the instruction level and the results are subsequently compared to detect a possible error. A program is run twice (or more at the task level to minimize complex faults. While this technique does not introduce additional hardware costs, it increases the time taken to ensure redundancy. The method reduces the efficiency of computation and thus absorbs more resources.

E. *Quality of IoT Service*

1) *Performance:*

How a system is going to perform in different scenarios by considering some set of measures like time constraints, environment, etc. In paper [12], distributed edge computational network is being discussed in which, by introducing preprocessing at end nodes, much of the computation is handled at the end nodes itself, hence reducing network traffic in between end nodes and the central server. This also helped in reducing network utilization and decreasing network latency.

2) *Availability & Security:*

Availability is the ability of the system, to be completely or partially working whenever required. Fault Tolerance and availability are not equivalent, as a fault-tolerant system is expected to keep the system running without interruption, however service interruptions can occur in a highly available system. A fault-tolerant scheme, however, should also preserve a high degree of device availability and performance.

Security is a major concern in IoT systems that link various components and entities through a network to each other. Paper [1] focused on introducing fault tolerance at home security systems, which detects home intrusions by making use of multiple devices.

3) *Scalability:*

As IoT systems should be able to work properly considering a large number of heterogeneous devices, scalability [9] is also an important attribute. It is difficult to comment on IoT scalability as a whole system, but it depends on how to incorporate new resources on demand.

4) *Interoperability:*

Interoperability allows heterogeneous IoT components to work efficiently together. The paper [4] performs a comprehensive survey on the state-of-the-art solutions for facilitating interoperability between different IoT platforms. Also, the key challenges in this topic are presented.

5) *Energy Consumption:*

Most IoT devices are battery-powered, and it is important to have energy efficiency linked to many other quality attributes, such as performance. Paper [9] introduces an algorithm to search disjoint paths that can minimize energy consumption while dealing with network link failure.

IV. COMPARISONS

The following section compares paper on the basis of architecture used, techniques employed, QoS achieved while carrying out fault tolerance in IoT based systems. The comparison is totally based on the study that is done on the following research papers by considering the different attributes. Crux of this section presented as current trends is discussed in subsequent section V.

A. *Architecture Layers:*

Our study shows that efficiency and availability are related to fault-tolerance of IoT systems. However the assessment of the trade-off between FT and other attributes of IoT efficiency, such as scalability, interoperability and energy consumption, will be further investigated. Another outcome to be further examined by an overview of the state of practice is that only a few studies facilitate the relationship between FT techniques and collaborative architects. All the paper considered falls on the aforementioned four architectural layers. Papers[1],[10],[11] are focused on the actuate layer, papers [1],[3],[9],[10],[11],[13] are related to sense layer, papers [4],[8],[10],[11] are based on processing and storage layer and papers [1],[8],[9], are focused on network layer**.** So different layers are being targeted in each paper to make a network fault tolerant.

B. *Architectural Patterns*

The question here is for each Fault Tolerance technique, which architectural pattern is more frequently used? Hybrid patterns [4] were used by studies to promote their passive Fault Tolerance techniques, while hybrids were used for active FT. Conversely, to deal with passive Fault Tolerance, unified and collaborative architectural patterns [10] are more fitting. Obviously, it is easier to approach the network control Fault Tolerance technique via a hybrid architectural pattern. In general, FT-IoT is assured by a hybrid architecture that if one fog node fails, the IoT device will move the computation to another fog to prevent a single point of failure. To achieve a fault tolerant network in paper [10] distributed collaborative pattern is followed, in papers [13], [11] centralized pattern is being employed, in papers [4],[8],[9] hybrid architecture is being implemented.

C. *Architectural Styles*

Different architectural styles followed in different papers to achieve a fault tolerant system. Styles employed in papers under study are as follows. Microservices style is used in paper [13], in papers [4],[10],[11],[5] service oriented style is being used, in paper[8],[4] cloud based architecture style is being used, in papers [8],[4],[11] layered style approach is being followed, in [8] publish/subscribe style is being used.

D. *Fault-Tolerance Techniques*

As mentioned in section III, to make a system fault tolerant different fault tolerant techniques can be employed. The different techniques used by different papers are described below.

1) *Replication:*

In papers [4], [8] active replication is being employed whereas in paper [10],[11],[12] passive replication is being employed.

2) *Network Control:*

The key studies have suggested many cluster-based routing protocols. Network control scheme is being employed in papers [10], [11].

3) Distributed Recovery Block:

In papers [3],[13] distributed recovery blocks technique is employed to ensure node computations are error free.

4) Time Redundancy:

In paper [3] time redundancy technique is followed.

E. Quality of IoT Service

Quality of Service (QoS) is also an ever-increasing network requirement today. New applications, such as voice and live video transmissions, which are accessible to consumers over the internet, generate higher standards for the quality of the services offered. When the traffic volume is greater than what can be transmitted over the network, devices queue, or keep, the packets are held in memory before the resources are made available to transmit them. In papers [8],[9],[11],[12] performance is focused. Availability is used as an attribute in papers[8],[9],[10], security is described in papers[1],[4], scalability is used as an attribute in paper[9], interoperability is used as an attribute in paper[4], energy consumption is focussed in paper[9].

V. TRENDS

It was observed that in most of the papers reviewed, in order to introduce fault tolerance actuate and sense layer was being targeted, replication and network control techniques were primarily employed, performance and availability is mostly discussed under QoS attribute. Some papers [1],[3],[9] discussed novel approaches to achieve fault tolerance techniques, removing disadvantages of old techniques.

Energy consumption, one of the QoS attributes is less focused while making system fault tolerant, so there is a wide scope in this field to work upon. Also, time redundancy technique is employed in a handful of papers, hence paving the way for future research. It was also observed that correspondence between fault tolerance techniques and associated architecture is less studied. So, despite fault tolerance in IoT being studied over a decade, there is still much scope of improvements in the field.

VI. CONCLUSION

In this paper, we present a systematic analysis of mapping with the objective of classifying and defining the state-of-the-art domain and extracting a collection of methods and techniques for Fault Tolerance in IoT. The fault tolerance capability of some papers shows that cloud data center faults can be addressed in real time by customized design before repair becomes available. In comparison to some of the contributions discussed in the related works, the transition of data from failed devices to safe ones takes more excessive time and causes delay. The findings of this study are both research-oriented and industry-oriented and are intended to establish a context for future Fault Tolerance IoT related research. We will analyses the possible incorporation of existing research at the industrial level of IoT as a future task. The study will help the readers to analyses the IoT system very minutely.

REFERENCES

[1] D. Terry, "Toward a New Approach to IoT Fault Tolerance," in Computer, vol. 49, no. 8, pp. 80-83, Aug. 2016.

[2] “What Is Fault Tolerance?: Creating a Fault Tolerant System: Imperva.” Learning Center, Imperva, 30 Dec. 2019, www.imperva.com/learn/availability/fault-tolerance/.

[3] Tusher Chakraborty, Akshay Uttama Nambi, Ranveer Chandra, Rahul Sharma, Manohar Swaminathan, Zerina Kapetanovic, and Jonathan Appavoo. 2018. Fall-curve: A novel primitive for IoT Fault Detection and Isolation. In Proceedings of the 16th ACM Conference on Embedded Networked Sensor Systems (SenSys '18). Association for Computing Machinery, New York, NY, USA, 95–107.

[4] N. Mohamed, J. Al-Jaroodi and I. Jawhar, "Towards Fault Tolerant Fog Computing for IoT-Based Smart City Applications," 2019 IEEE 9th Annual Computing and Communication Workshop and Conference (CCWC), Las Vegas, NV, USA, 2019, pp. 0752-0757.

[5] Gubbi, Jayavardhana, Rajkumar Buyya, Slaven Marusic, and Marimuthu Palaniswami. "Internet of Things (IoT): A vision, architectural elements, and future directions." Future generation computer systems 29, no. 7 (2013): 1645-1660.

[6] Moghaddam, Mahyar Tourchi, and Henry Muccini. "Fault-tolerant iot." In International Workshop on Software Engineering for Resilient Systems, pp. 67-84. Springer, Cham, 2019.

[7] Rullo, Antonino, Edoardo Serra, and Jorge Lobo. "Redundancy as a Measure of Fault-Tolerance for the Internet of Things: A Review." In Policy-Based Autonomic Data Governance, pp. 202-226. Springer, Cham, 2019.

[8] A. Javed, K. Heljanko, A. Buda and K. Främling, "CEFIoT: A fault-tolerant IoT architecture for edge and cloud," 2018 IEEE 4th World Forum on Internet of Things (WF-IoT), Singapore, 2018, pp. 813-818

[9] M. Z. Hasan and F. Al-Turjman, "Optimizing Multipath Routing With Guaranteed Fault Tolerance in Internet of Things," in IEEE Sensors Journal, vol. 17, no. 19, pp. 6463-6473, 1 Oct.1, 2017.

[10] P. H. Su, C. Shih, J. Y. Hsu, K. Lin and Y. Wang, "Decentralized fault tolerance mechanism for intelligent IoT/M2M middleware," 2014 IEEE World Forum on Internet of Things (WF-IoT), Seoul, 2014, pp. 45-50.

[11] S. Zhou, K. Lin, J. Na, C. Chuang and C. Shih, "Supporting Service Adaptation in Fault Tolerant Internet of Things," 2015 IEEE 8th International Conference on Service-Oriented Computing and Applications (SOCA), Rome, 2015, pp. 65-72.

[12] M. Mudassar, Y. Zhai, L. Liao and J. Shen, "A Decentralized Latency-Aware Task Allocation and Group Formation Approach With Fault Tolerance for IoT Applications," in IEEE Access, vol. 8, pp. 4912-4923,2020.

[13] A. Celesti, L. Carnevale, A. Galletta, M. Fazio and M. Villari, "A Watchdog Service Making Container-Based Micro-services Reliable in IoT Clouds," 2017 IEEE 5th International Conference on Future Internet of Things and Cloud (FiCloud), Prague, 2017, pp. 372-378.