Disaster-FD: A Failure Detector for Disaster-Prone Environments

Abadio de Paulo Silva
Federal University of Uberlândia
Uberlândia, Brazil
abadiops@ufu.br

Anubis Graciela de M. Rossetto
Sul-Rio-Grandense Federal Institute
Passo Fundo, Brazil
anubisrossetto@ifsul.edu.br

Pierre Sens
Sorbonne University
Paris, France
pierre.sens@lip6.fr

Luciana Arantes
Sorbonne University
Paris, France
luciana.arantes@lip6.fr

Rafael Pasquini
Federal University of Uberlândia
Uberlândia, Brazil
rafael.pasquini@ufu.br

Paulo Coelho
Federal University of Uberlândia
Uberlândia, Brazil
paulocoelho@ufu.br

Abstract

This paper introduces Disaster-FD, a failure detector designed for disaster-prone environments, focusing on real-time IoT network monitoring. Inspired by Impact-FD, Disaster-FD features active and federated monitoring to ensure network reliability under adverse conditions. Tested on the IoT-LAB platform, Disaster-FD demonstrates robust performance, enhancing IoT network resilience during disasters. It assesses reliability thresholds, confidence levels, and impact factors, ensuring efficient energy consumption and maintaining high network trust. This paper details Disaster-FD's implementation, performance metrics, and experimental results, highlighting its potential for effective support in disaster management.

CCS Concepts

• **Computer systems organization** → **Fault-tolerant network topologies**; **Reliability**; • **Networks** → *Sensor networks*; • **Computing methodologies** → **Distributed algorithms**.

Keywords

IoT, Disaster Management, Failure Detectors, Trust Level

ACM Reference Format:
Abadio de Paulo Silva, Anubis Graciela de M. Rossetto, Pierre Sens, Luciana Arantes, Rafael Pasquini, and Paulo Coelho. 2024. Disaster-FD: A Failure Detector for Disaster-Prone Environments. In *13th Latin-American Symposium on Dependable and Secure Computing (LADC 2024), November 26–29, 2024, Recife, Brazil.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3697090.3697096

1 Introduction

Natural disasters represent one of the greatest threats to contemporary society, causing devastating impacts ranging from disrupting essential services such as water and power supply to significant economic losses, damage to public and private properties, and most critically, the loss of human lives. These events afflict the world and, particularly in Brazil, most natural disasters are climate-related.

According to a report by the National Confederation of Municipalities (CNM) in 2022, about 3,400 people in Brazil were directly affected by natural disasters [11]. This challenging context underscores the urgent need to develop effective strategies for monitoring and managing risks associated with these disasters. A promising approach to address this challenge is to leverage remote monitoring technologies based on the Internet of Things (IoT). The interconnectivity of various objects in a network, provided by the IoT paradigm, offers a unique opportunity to enhance disaster response.

However, the effectiveness of this strategy is often compromised in large-scale disaster scenarios, where communication infrastructure, including IoT devices installed for monitoring, may fail. Such failure can underestimate the severity of the disaster due to a lack of data and prevent the issuance of critical alarms. Thus, it becomes essential to develop robust algorithms capable of efficiently monitoring an IoT network, ensuring device availability, and establishing a level of trust in the processes originating from this ecosystem. Additionally, it is crucial to propose improvements in the management of these sensors to ensure efficient and reliable monitoring, even under adverse conditions.

In such scenarios, the importance of effective failure detectors becomes evident. The work in [6], a pioneer in the study of unreliable failure detectors, highlights the importance of fundamental properties such as completeness and accuracy in such systems. These properties ensure that algorithms using failure detectors maintain consistency in their decisions and do not become indefinitely blocked [5].

This article is in the context of STIC-AMSud ADMITS project, a cooperation with universities in Brazil, France, Uruguay, and Chile, and aims to develop an algorithm for real-time monitoring of Internet of Things networks. Inspired by the *Impact Failure Detector* [14], or *Impact-FD*, the new proposed algorithm extends the latter by adding certain features, such as active and federated monitoring of IoT devices. The main goal is to monitor the availability of IoT devices and establish a level of reliability for the network, considering a specific set of monitored processes. The proposal also evaluates the impact of Disaster-FD on the energy consumption of IoT devices in the network.

The IoT-LAB [1], with over 1500 sensor nodes spread across various locations in France, including Grenoble, Lille, Saclay, and Strasbourg, stands out as one of the largest open testbeds available

LADC 2024, November 26–29, 2024, Recife, Brazil

ACM ISBN 979-8-4007-1740-6/24/11
https://doi.org/10.1145/3697090.3697096

to the international scientific community. To facilitate experimentation in complex IoT networks, the IoT-LAB supports a variety of communication protocols.

Additionally, the IoT-LAB enables federated experiences. This means that devices can be programmed and managed simultaneously across multiple regions, providing an ideal platform for testing algorithms and distributed applications in a network that simulates the complexity of the global Internet.

The concept of federated monitoring in IoT networks refers to the practice of integrating and managing multiple autonomous IoT device networks that are geographically distributed or belong to different administrative domains. This monitoring model is designed to enhance the efficiency, security, and resilience of large IoT systems, providing more effective supervision and a coordinated response to incidents or failures. By monitoring different regions simultaneously, it is possible to quickly identify when an area begins to deteriorate due, for instance, to failures or anomalies. Such a detection is crucial in critical infrastructures, such as power networks, where a failure in one area can cascade to other regions.

In a federated system, each IoT network operates independently but shares information with other networks to improve global surveillance and management. This approach is emphasized by the survey in [3], where authors discuss the need for collaboration among devices and distributed systems.

The remainder of the paper is organized as follows. Section 2 introduces the system model and associated definitions. Section 3 describes Disaster-FD. Section 4 discusses the estimated arrival calculation. Section 5 details the implementation, with the experimental evaluation in Section 6. Section 7 briefly compares Disaster-FD to related work, and Section 8 concludes the paper.

2 System Model and Definitions

This section presents the system model and definitions related to the scope of the work.

2.1 System Model

This work considers a distributed system consisting of a finite set of processes $\Pi = q_1, \ldots, q_n$, where $|\Pi| = n$, $(n \geq 2)$, and the existence of only one process per node or sensor. Each node (or process) has a unique identifier. The identifiers are consecutively ordered. Processes can fail by crashing and do not recover. A process is considered correct if it does not fail throughout the entire execution.

This distributed system is **asynchronous**, defined as a system where there is no bound on the transmission time for messages or execution time for a processing step, meaning there are no assumptions related to timing [8, 16]. In this type of system, no mechanism can guarantee the failure of a remote process, as it is impossible to differentiate a failed process from one that is merely slow or experiencing slow communication.

The system has **fair lossy** communication channel. According to [2], a lossy channel satisfies the property of integrity, meaning that a process q receives a message m from another process p at most once, and only if p previously sent m to q. Practically, this means that messages cannot be created spontaneously, and if a message m is not lost, it is eventually received at its destination. Additionally, channels can intermittently drop messages. Fairness of losses requires that if a correct process p sends a message m an infinite number of times then the channel will deliver m an infinite number of times.

2.2 Unreliable Failure Detectors

Fault detection is a key element for ensuring the reliability and stability of distributed systems, especially asynchronous ones. Failure detectors can be used to circumvent the impossibility of the FLP (Fischer, Lynch, and Patterson) theorem [5, 10], which shows that in an asynchronous setting, where only one process might crash, no distributed algorithm solves the consensus problem [4, 13] deterministically.

The work in [6] introduced the concept of unreliable failure detectors, defined by two fundamental properties: completeness and accuracy. **Completeness** characterizes the failure detector's capability of suspecting faulty processes, while **accuracy** characterizes the failure detector's capability of not suspecting correct processes, i.e., restricts the mistakes that the failure detector can make. In practice, these failure detectors produce an output list of processes considered suspicious.

Disaster-FD, proposed in this paper, extends the definition of failure detectors, using the same approach defined in [14]. In this context, there is a process $p \in \Pi$ that monitors a subset S of Π. Each process in S connects to p via a communication channel.

Thus, unlike traditional detectors defined in [6], Disaster-FD, similarly to Impact-FD, can be defined as an unreliable failure detector that provides output related to the **confidence level** in the processes in S. If the confidence level is equal to or higher than a **threshold** defined by the user, then the system is considered reliable. The work in [14] discusses the equivalence between such concepts and the definitions in [6].

3 Disaster-FD

By extending Impact-FD [14], Disaster-FD is a failure detector conceived for disaster-prone environments with multiple regions that provides intra and inter-region real-time monitoring.

Disaster-FD introduces the use of multiple monitor processes, each deployed in a monitored region, as illustrated in Figure 1. In this scenario, each region has a monitor process and a set of IoT devices. The monitor of each region observes the devices in its region and a subset of the processes in another region. Such an arrangement enhances fault detection by preventing the complete failure of a region from going unnoticed.

In this context, each process q within the subset $S \subset \Pi$ is assigned an impact factor. This factor, a positive integer value, reflects the relative importance of the process within the system. For example, in the scenario shown in Figure 1, a higher impact factor may be assigned to monitors compared to the impact factor of sensors, indicating that the failure of a monitor has a stronger consequence on the good operation of the system than the failure of a sensor.

Each monitor process p executes the Disaster-FD algorithm, i.e., regularly assesses the connectivity and calculates the confidence level for each monitored process in the set S, thereby establishing the overall system reliability. This confidence level is determined by the sum of the impact factors of the processes that, at that moment, are not considered faulty.

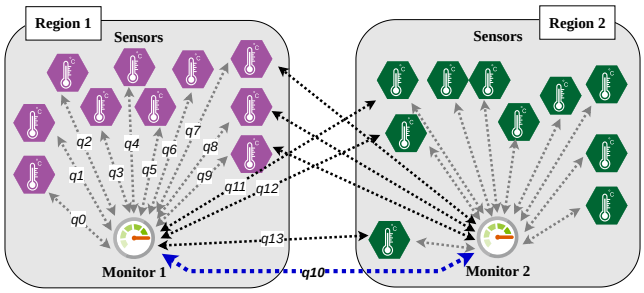

Figure 1: Example of a monitoring scenario with 2 regions and 10 sensors.

In addition, a threshold must define the minimum level of reliability required by the system. Consequently, the values of the impact factor must be chosen in such a way that the set of S processes controlled by each regional monitor gives a confidence level below the threshold whenever the minimum reliability level is not reached.

3.1 Formalization

This section formalizes the definitions and concepts used in this paper. Disaster-FD follows the same nomenclature adopted in [14], expanding the concepts to a multi-monitoring or federated monitoring scenario. This incorporation enhances the system's efficiency, accuracy, and adaptability, essential requirements for addressing the uncertainties and rapid changes in disaster scenarios.

The **Impact Factor** assigned to each process corresponds to a positive integer value that indicates its relative importance in the system. The impact factor of each monitored process i, (I_i), along with the unique identifier of the process, make up the set $S \subset \Pi$ of monitored processes. Thus, the values in the set S correspond to the set $\langle id_1, I_1\rangle, \langle id_2, I_2\rangle, \ldots, \langle id_k, I_k\rangle$ for each process $i \in S, 1 \leq i \leq k$.

For each monitored set S, the subset $T_p^S(t)$ represents the processes that the monitor p suspects at the time t. Complementarily, the set $F_p^S(t)$ represents the processes considered faulty by the monitor p at the moment t.

The **Trust Level** indicates the confidence level of the monitor process p in the set of processes in S at a given moment, calculated by $TL_p^S(t)$. It represents the sum of the impact factors of the non-faulty processes, that is, $TL_p^S(t) = \sum_i (I_i), \forall i \in T_p^S(t)$.

Each monitor can track various subsets of processes, with different levels of trust and individual impact factors. The set S^* encompasses the m unique subsets being monitored, indicated by $S^* = \{S_1, S_2, S_3, \ldots, S_m\}$.

Finally, the **Threshold** defines the minimum reliability limit for each set in S^*, mathematically represented by $\{Th_1, Th_2, \ldots, Th_m\}$, where each Th_i is related to the minimum level of trust required for a subset of processes S_i.

The values in Th_{S^*} are used by the monitor to verify the trust in the processes of the subsets in S^*. If, for each of the m subsets of $S^*, 1 \leq i \leq m$, the $TL_p^i(t) > Th_i$, then S^* is considered reliable (*trusted*) at time t by the monitor p; otherwise, S^* is considered not reliable (*not trusted*).

This class of algorithms introduces the concept of **Flexibility Property**, which denotes the failure detector's ability to tolerate a certain margin of failures or false suspicions, that is, its ability to consider different sets of responses that lead the system to states to be considered trusted.

Table 1: Set S_1 with processes q_i and respective impact factors.

Set S_1 monitored by the monitor process of Region 1
$\langle q_0, 10\rangle, \langle q_1, 10\rangle, \langle q_2, 10\rangle, \langle q_3, 10\rangle, \langle q_4, 10\rangle, \langle q_5, 10\rangle, \langle q_6, 10\rangle,$ $\langle q_7, 10\rangle, \langle q_8, 10\rangle, \langle q_9, 10\rangle, \langle q_{10}, 60\rangle, \langle q_{11}, 20\rangle, \langle q_{12}, 20\rangle, \langle q_{13}, 20\rangle$

Table 1 presents an example with a set of monitored processes similar to the scenario depicted in Figure 1. The set S_1 of the monitor in Region 1 comprises the processes q_0 to q_9 representing sensors located in Region 1 (purple), the processes q_{11} to q_{13} representing remote sensors under monitoring (green), and q_{10} as the monitor of Region 2. The maximum value of $TL_p^{S_1}(t)$ is $\sum_{i=0}^{13} I_i = 220, \forall t > 0$, with $\sum_{i=0}^{9} I_i = 100$ for local sensors and $\sum_{i=10}^{13} I_i = 120$ for remote sensors and monitor. In this situation, the chosen threshold should reflect the monitoring objective. For example, to ensure that at least one process from each region always responds, it is necessary to have $120 < Th_1 \leq 220$ and $TL_p^{S_1}(t) > Th_1, \forall t > 0$. The monitor of Region 2 can adopt an equivalent strategy.

3.2 Quality of Service (QoS) Metrics

The evaluation of Disaster-FD is based on the work by [7], where they define a set of metrics to assess the Quality of Service (QoS) of fault detection algorithms. These metrics are centered around temporal constraints, which refer to the time required to detect a failure, to correct a mistake, and the interval between two false suspicions. Specifically, the same QoS metrics used by Impact-FD are adopted:

- **Average Detection Time** (TD): Measures the speed and efficiency of the system in detecting failures. TD measures the period from the moment a process q fails until the failure detector in p begins to continuously suspect q. This metric is critical for understanding how quickly the system responds to incidents.
- **Average Error Rate** (μR): Represents the frequency at which the failure detector makes errors per unit of time, serving as an indicator of the detector's reliability. This metric is particularly important for assessing the system's propensity for false positives or false negatives.
- **Accuracy Probability** (PA): Assesses the likelihood that the outputs of the failure detector are correct at a random moment, providing a measure of the overall accuracy of the system over time, derived from the total duration of the false positive period relative to the total time under analysis.

4 Estimation of Heartbeat Arrival

The basic monitoring mechanism consists of receiving periodic messages from the monitored processes, commonly referred to as **heartbeats** (HB). The method proposed by [7] to estimate the arrival of the next heartbeat (EA_{k+1}) is based on the history of the

arrival times of previous heartbeats and includes a safety margin (β).

In the EA_{k+1} calculation, the process p considers a sliding window with the w most recent heartbeat messages received from process q represented by $m_1, m_2, ..., m_w$. The values $T_1, T_2, ..., T_w$ are the respective reception times of these messages, according to the local clock of p. Thus, as defined in [7], we have Equation 1, where Δ_i corresponds to the interval between the sending of two consecutive heartbeats.

$$EA_{k+1} = \frac{1}{w} \sum_{i=k-w}^{k} (T_i - \Delta_i \times i) + (k+1) \times \Delta_i \quad (1)$$

Thus, the expected arrival time of heartbeat $k+1$ called τ_{k+1} is defined by Equation 2, and the non-receipt of a heartbeat by the time τ_{k+1} characterizes process q as suspect.

$$\tau_{k+1} = EA_{k+1} + \beta \quad (2)$$

4.1 Comparison with Impact-FD

While both Disaster-FD and Impact-FD use Equations 1 and 2 to calculate the estimated time of arrival of the next heartbeat, named EA_{k+1}, the two proposals use slightly different approaches in interpreting and implementing the Equation 1.

The implementation of the Disaster-FD protocol uses the sequence number identifier of the heartbeat message to calculate the difference between the actual arrival time and a theoretical arrival time, defined in Equation 2 as the product of the identifier by the fixed interval between consecutive heartbeats (Δ_i). The Impact-FD protocol, on the other hand, does not directly use the sequence number of the heartbeat, employing an incremental index to calculate the difference between the arrival time of each heartbeat and the expected arrival time, based on Δ_i.

The methodology proposed by Chen is based on adjusting the arrival time estimate using the history of arrival times of the previous w messages, considering the difference between the actual and expected arrival times, based on the regular interval between heartbeats.

Thus, Disaster-FD is more aligned with Chen's theory, as it directly incorporates the concept of heartbeat sequentially (through the sequence number), reflecting Chen's approach of adjusting estimates based on differences between actual and expected arrival times. Although similar in structure, by using an incremental index, the Impact-FD does not always capture sequentiality directly, and thus may not accurately have the actual sequence of heartbeats.

In practice, as observed when analyzing the logs of Impact-FD experiments, this means that Impact-FD has more difficulty handling "gaps" in a sequence of heartbeats, which occur when some messages are lost. As a result, the Impact-FD implementation requires more time to detect a false positive, i.e., to realize that it has erroneously suspected a correct process.

5 Implementation

Disaster-FD has been implemented in Java and utilizes the Californium library [12]. It is distinguished by its federated and multi-protocol monitoring, using CoAP (Constrained Application Protocol) requests for monitoring IoT devices and the ICMP (Internet Control Message Protocol) to monitor nodes of neighboring federated regions. This dual approach provides flexibility and a more comprehensive network state analysis.

CoAP "CON" (Confirmable) requests enhance communication reliability, as each message anticipates a response from the destination device. Furthermore, the system implements a timeout mechanism for these requests, ensuring that the monitoring remains efficient even when a device does not respond within the expected time. Disaster-FD uses the Californium library to handle responses from CoAP requests asynchronously, managing successful returns of heartbeat messages as well as handling errors, such as the overflow of the calculated time for the next heartbeat reception and connectivity issues, allowing continuous monitoring and uninterrupted analysis flow.

5.1 Implementation of CoAP Message Tracking

Initially, each IoT device is accessible via a unique URI (Uniform Resource Identifier), based on its specific IPv6 address. Using this URI, the CoAP client in the Disaster-FD implementation derives a unique device identifier, used in generating the initial value of the message sequence number, or MID (Message ID).

The MID is generated or retrieved for each device and incremented with each new request, ensuring the uniqueness and traceability of messages. In CoAP GET and CON requests, the MID is explicitly defined in the header, allowing for a precise correlation between the sent requests and the received responses.

6 Results

This study implements and tests the Disaster-FD failure detection system in the Internet of Things (IoT) environment known as FIT-IoTLAB [1], monitoring two interconnected regions, Grenoble and Strasbourg, in a configuration similar to the example in Figure 1. The choice of only these two regions is justified by connectivity issues between the remaining regions within the infrastructure, likely related to firewall configurations.

Table 2 shows the connectivity results of a simple experiment where we deploy IoT devices in every region and issue ICMP requests from each monitor node to every other available region. The results show that only Strasbourg and Grenoble have full bidirectional connectivity.

Table 2: Connectivity between pairs of FIT-IoTLAB regions.

From/To	**Grenoble**	**Lille**	**Saclay**	**Strasbourg**
Grenoble	X			X
Lille	X	X		X
Saclay	X		X	X
Strasbourg	X			X

Experiments rationale

The experiments were structured around the monitoring of 14 processes by each region monitor, including 10 sensors located in the monitor's region and 3 sensors and one monitor process

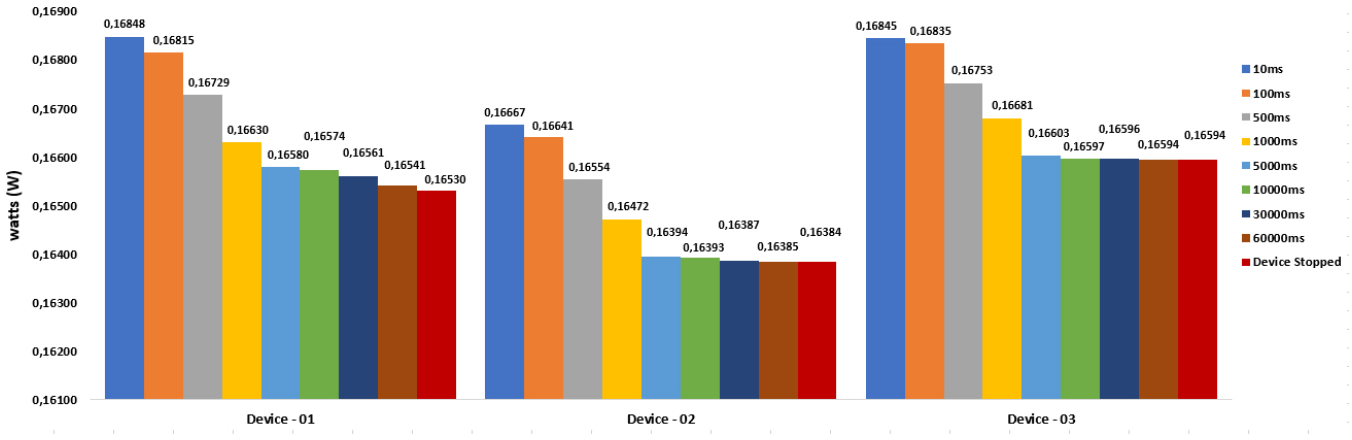

Figure 2: Energy consumption with different request intervals.

in the other region. Each monitor uses the CoAP protocol for the sensors and ICMP to track other monitors.

Requests to the sensors were made using the GET method of the CoAP protocol at an interval of 5,000 milliseconds. This rate was carefully selected aiming for efficiency in fault detection, minimization of network load, and energy consumption on devices, as shown in § 6.1.

The choice of the β margin in calculating the estimated time for the next heartbeat was also defined based on latency tests of sending and receiving messages to devices in the two chosen regions. After 24 hours of testing, a value corresponding to the calculated standard deviation was chosen, amounting to 1,500 milliseconds.

Such experiments aim not only to evaluate the effectiveness of Disaster-FD in detecting faults in real-time in an IoT environment but also to explore the interactions between network devices in federated regions. The setting of the impact factors in each region was defined similarly to that presented in Table 1, namely: the 10 sensors located in the local area of each region were set with an impact factor of 10. In comparison, the 3 sensors in the neighboring region received an impact factor of 20. Additionally, the monitor in the neighboring region has an assigned impact factor of 60, reflecting its importance in that area. Consequently, the trust level for this set of processes can reach a maximum of 220 ($10 \times 10 + 3 \times 20 + 60$).

The experiments evaluate the impact of Disaster-FD on IoT device power consumption, the individual device accumulated errors, the accuracy of the estimated arrival time, as well as overall system behavior based on the properties defined in Section 3.2 for Disaster-FD and Impact-FD.

6.1 Energy consumption for IoT devices

Our goal is to investigate the impact of energy consumption on IoT devices within request sending intervals. Using the IoT-LAB platform, experiments were conducted in three distinct devices, each configured to issue GET requests via the CoAP protocol.

Figure 2 illustrates energy consumption in watts (W) for three IoT devices at different sending intervals, varying from 10 milliseconds to 60,000 milliseconds. The figure shows a greater consumption discrepancy between 10 milliseconds and 1,000 milliseconds sending interval. On the other hand, it is observed that, from 5,000 milliseconds onwards, energy consumption presents a tendency to stabilize, similar to the values observed when the devices are not receiving any CoAP request ("device stopped").

When expanding the experiments to longer intervals, such as 10,000 milliseconds, 30,000 milliseconds, and 60,000 milliseconds, we observe that the power consumption remains close to the 5,000 millisecond scenario. However, these larger intervals can be counterproductive in contexts that require agile responses and frequent data updates.

Evaluation results show that a sending interval of 5,000 milliseconds represents a good trade-off, mitigating energy consumption without compromising the frequency of updates. This sending interval represents an economy in energy consumption of 0.03 W, or 2%, compared to a 10-millisecond sending interval for each device. Even though it represents a small energy save, this can be crucial for devices operating in environments where access to power is restricted or battery replacement is unfeasible.

6.2 Accumulated Errors in Strasbourg

Figure 3 shows the accumulated number of errors that occurred per monitored device during the 24 hours of monitoring in the Strasbourg region.

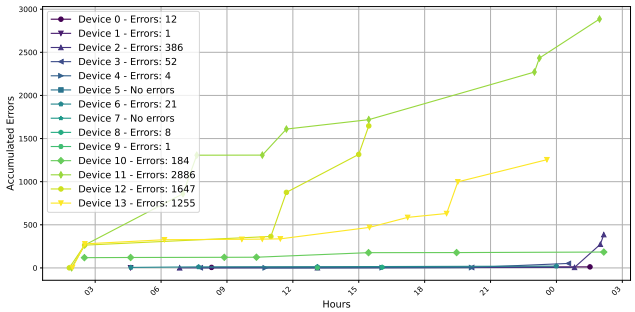

Figure 3: Accumulated Errors per device in Strasbourg.

Devices 0 to 9 correspond to local sensors, devices 11 to 13 correspond to sensors in the Grenoble region, and device 10 represents the monitor in Grenoble.

The detector did not register any failures for devices 5 and 7, while the IoT devices 0, 1, 2, 3, 4, 6, 8, and 9 showed only a few errors, reflecting the stability of the Strasbourg region. In contrast, in the neighboring region of Grenoble, both the monitored IoT devices (11, 12, and 13) and the monitor (device 10) showed a significant number of errors, indicating instability in that area.

Notably, device 10, which serves as the monitor in the Grenoble region, registered several errors. This device also acts as the central node or edge router and is crucial for managing network traffic in Grenoble. The errors observed in device 10 suggest connectivity problems, compromising communication and the operational efficiency of the region's IoT network. Therefore, the instability of device 10 is a critical factor that can affect the performance of the Grenoble network, causing interruptions in functionality and failures to respond to requests sent by the Disaster-FD monitor.

6.3 Accumulated Errors in Grenoble

In a similar manner to the previous section, Figure 4 shows the accumulated number of errors for each monitored device over 24 hours in the Grenoble region. Devices 0 to 9 correspond to local sensors, devices 11 to 13 correspond to sensors in the Strasbourg region, and device 10 represents the monitor in the Strasbourg region.

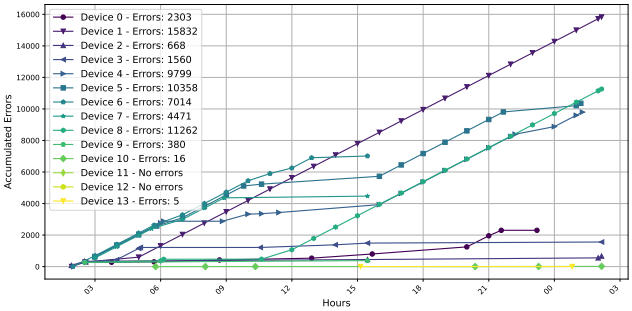

Figure 4: Accumulated errors per device in Grenoble.

The results provided by the Disaster-FD failure detector reinforce the observations in the previous section and Figure 3, especially regarding the network instability in the Grenoble region. While the Strasbourg monitor identified instability in Grenoble, the monitoring in Grenoble also detected a high incidence of errors in its own devices (devices 0 to 9) in Figure 4 and few in Strasbourg (devices 10 to 13) in Figure 4, corroborating the mutual perception of network performance between the two regions.

6.4 Actual and Estimated Arrival Times in Strasbourg

Figure 5 analyses the behavior of device 5, located in Strasbourg, during the 24-hour monitoring period. The choice of this device for analysis is justified by its operational robustness, as evidenced by its stable performance, which is also highlighted in Figure 3. The blue curve indicates the *arrival time* and corresponds to the difference between two consecutive heartbeats. The red curve indicates the *estimated arrival time*, i.e., the maximum time a heartbeat is expected to arrive before the device is considered faulty.

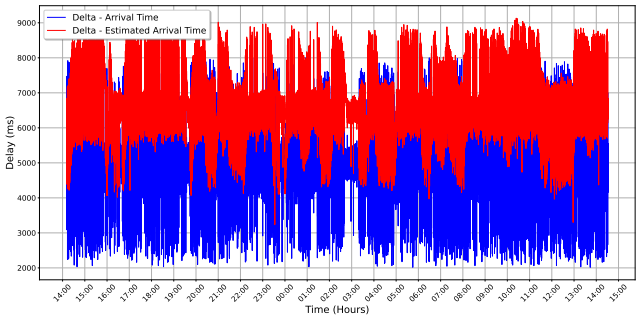

Figure 5: Arrival and Estimated Times for Device 5 in Strasbourg.

As discussed in the introduction of Section 6, the monitor adopted a safety margin of 1500 milliseconds and intervals of 5000 milliseconds for sending requests. A comparative analysis between the estimated arrival times (in red) and the actual heartbeats arrival times (in blue) revealed the stability of the system, demonstrated by the small variation in the time intervals between the estimate and the actual reception of the heartbeats. Additionally, the prevalence of red points over the blue points indicates that the heartbeats arrived at the monitor before the estimated time.

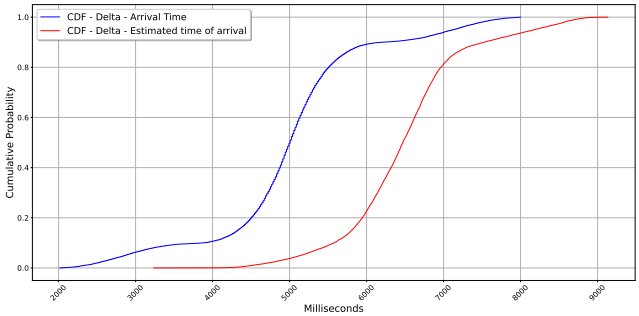

Figure 6: Comparison of Actual and Estimated Arrival Times in Strasbourg.

Figure 6 emphasizes the relation between actual and estimated message arrival times in Strasbourg, using a Cumulative Distribution Function (CDF) plot. The X-axis represents the time in milliseconds, while the Y-axis shows the cumulative probability. The blue curve indicates the actual arrival time of events, with 50% of events occurring before 5,000 milliseconds and 100% before 8,000 milliseconds. The red curve, representing the estimated time with a safety margin of 1,500 milliseconds, shows that the estimated times are greater than the actual times, showing the accuracy of the arrival time estimates and indicating a conservative approach in the estimates and the precision of the Disaster-FD system in predicting message arrival times.

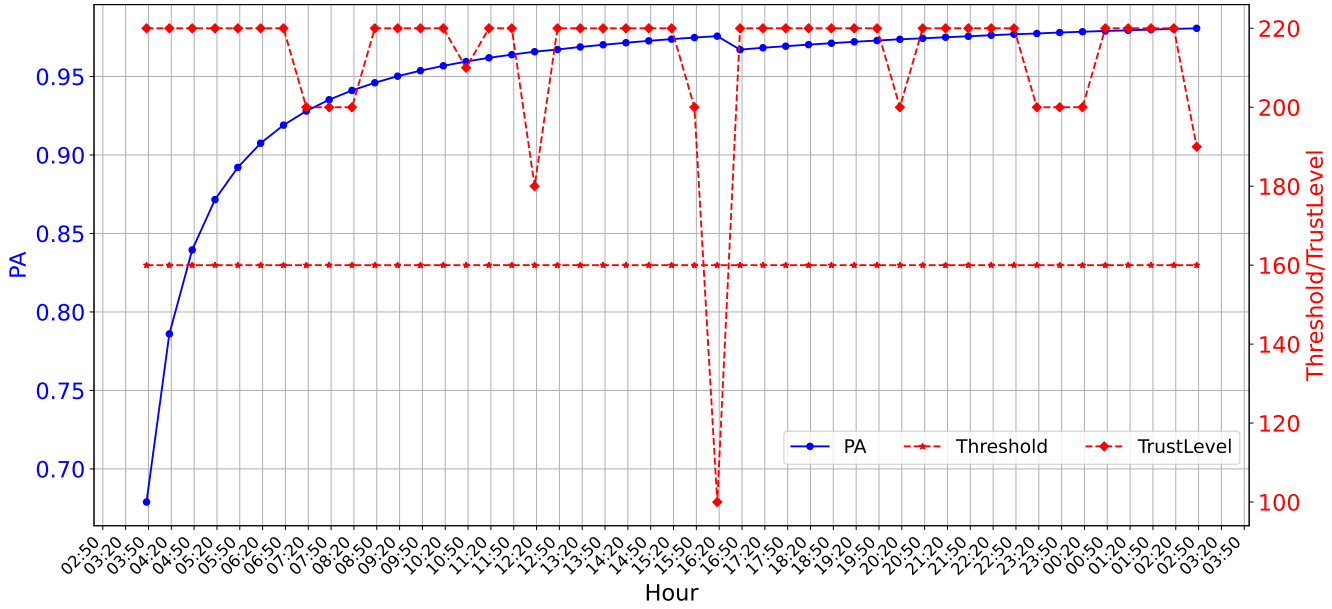

Figure 7: Network statistics in Strasbourg.

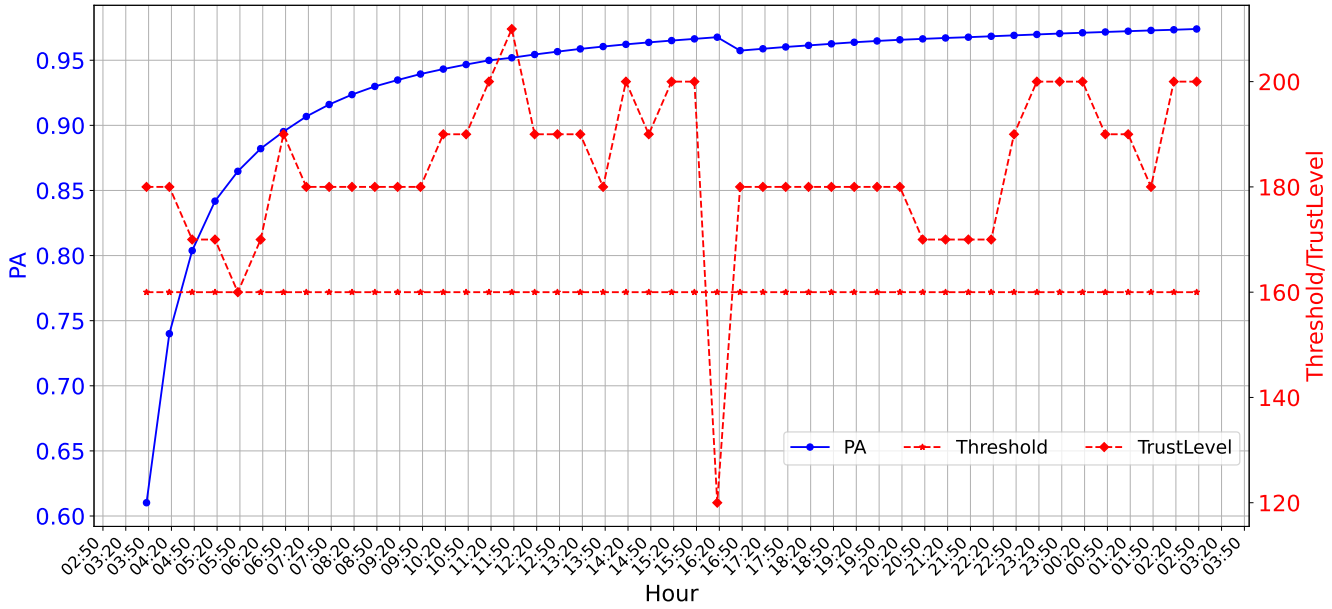

Figure 8: Network statistics in Grenoble.

6.5 Network Statistics in Strasbourg

The experiment was conducted under an established threshold value of 160 (dashed red line in Figure 7), which serves as a safety limit for the network in the Grenoble and Strasbourg regions, as detailed in the discussion of the values in Table 1 in Section 3.1. This value of 160 corresponds, for example, to situations where all nodes in Strasbourg are not suspected, and at least the monitor in Grenoble is responding.

The analysis, depicted in Figure 7, provides a statistical perspective on the network behavior regarding the trust level ($TL_p^S(t)$) (red curve).

Specifically, the trust level is influenced by the network stability parameters and, during the testing period, the accuracy of the

Table 3: Extract of the logs for the monitors in Strasbourg and Grenoble.

Monitor Region	Trust Level	Devices Trust Array	Probability of Accuracy	Timestamp
Strasbourg	100	T T T T T T T T T T F F F F	0.8061	04-01-2024 03:17
Strasbourg	100	T T T T T T T T T T F F F F	0.9756	04-01-2024 16:17
Grenoble	120	F F F F F F F F F F T T T T	0.9963	04-01-2024 03:17
Grenoble	120	F F F F F F F F F F T T T T	0.9676	04-01-2024 16:17

Disaster-FD system showed a progressive improvement as the monitored devices maintained stable operations.

Figure 7 also shows that the Probability of Accuracy (PA), identified by the blue curve, remains above 95% from 8:50 onwards, highlighting the monitor's ability to accurately indicate the system state.

Finally, it is noteworthy, from the analysis of the same figure, that the regions under analysis were considered unreliable between 15:50 and 16:50 when the trust level fell below the established threshold. In a real deployment, this situation could indicate a possible disaster, necessitating emergency actions for the affected regions.

Additionally, Table 3 presents two critical points extracted from the monitor's log in Strasbourg where the trust level recorded a value of 100, i.e., below the established threshold. Notably, the sequence of "Device Trust Array" 'TTTTTTTTTTFFFF' indicates a failure (value 'F') in the four devices of the neighboring network in Grenoble, devices 10 to 13, with impact factors of 60, 20, 20, and 20, respectively.

6.6 Network Statistics in Grenoble

Figure 8 corroborates the results presented in Figure 4, indicating that the network in the Grenoble region experienced periods of instability. This conclusion is evidenced by the analysis of the trust level ($TL_p^S(t)$) (in red), which at various times approached the established safety threshold for the network (dashed red line).

Furthermore, as shown in Table 3, there were failures in the local devices of Grenoble, resulting in an unreliable network classification. This can be observed by the value of the complementary "Devices Trust Array" of 'FFFFFFFFFFTTTT', shown in Table 3 for the same time in the Grenoble region, meaning the 10 devices in Grenoble are perceived as faulty ('F') by the region monitor, at the same time that the 3 devices and the monitor in Strasbourg remain trusted ('T').

6.7 Comparison with Impact-FD in Strasbourg

Figure 9 illustrates the statistics for Strasbourg with results for Impact-FD. The network statistics using the Impact-FD implementation of Equation 1, showed that the Probability of Accuracy (P.A) varied between approximately 0.40 and 0.90 over time. The average PA remained above the threshold of 0.40, initially lower than that of Disaster-FD, but indicating an improvement in failure detection accuracy throughout the experiment. The network safety threshold was maintained at 160. The trust level varied between 80 and 220, showing moments of instability during the period, especially around 15:50 to 16:50, when there was a significant drop in confidence. In a real scenario, this could indicate a possible disaster, triggering emergency actions for the affected regions, a scenario also presented by the Disaster-FD failure detector.

Comparing the network statistics for Strasbourg using the Impact-FD and Disaster-FD formulas, it is observed that Disaster-FD demonstrated a probability of accuracy varying between 0.70 and 0.95, reaching 0.95 early on, as observed in Figure 7. In contrast, Impact-FD showed greater variability, with the PA ranging between 0.40 and 0.90, reaching 0.90 only at the end of the experiment.

Thus, Disaster-FD was more stable and adapted quickly to network instabilities, while Impact-FD was less stable and adapted more slowly. Equation 1 implementation in Disaster-FD is responsible for the observed stability. Disaster-FD considers the message ID instead of a simple counter to calculate the estimated arrival time of the next message, contributing to greater accuracy and stability in failure detection.

6.8 Comparison with Impact-FD in Grenoble

Figure 10 illustrates the statistics for the Grenoble region using the Impact-FD implementation of Equation 1. For the Grenoble network, using the Impact-FD formula, the PA ranged from approximately 0.02 to 0.80. The network safety threshold was maintained at 160. The trust level varied between 120 and 200, with more pronounced instabilities throughout the experiment, which was expected due to the region's higher instability. The lower PA during critical moments indicated a lower reliability in failure detection, resulting in a higher occurrence of false positives.

When comparing the graphs for the Grenoble region, Figure 8 demonstrates that Disaster-FD maintained a higher and more stable accuracy (PA), ranging from 0.60 to 0.95. At the same time, Impact-FD showed greater variability, with PA ranging from 0.02 to 0.80. Grenoble experienced more network instabilities during the experiment compared to Strasbourg. Disaster-FD quickly adapted to these instabilities, unlike Impact-FD, which had slower adaptation and inconsistencies in detection. The safety threshold was maintained at 160 for both formulas.

The trust level in the Disaster-FD formula also exhibited a clear advantage in terms of stability and less variability. Using the message ID in the Disaster-FD formula resulted in a more accurate arrival time estimate, improving accuracy (P.A) and significantly reducing the probability of false positives.

7 Related Work

The field of failure detection in Internet of Things (IoT) networks has seen significant advancements, with several notable contributions aimed at improving the reliability and efficiency of such systems. This section reviews the literature on failure detection technologies,

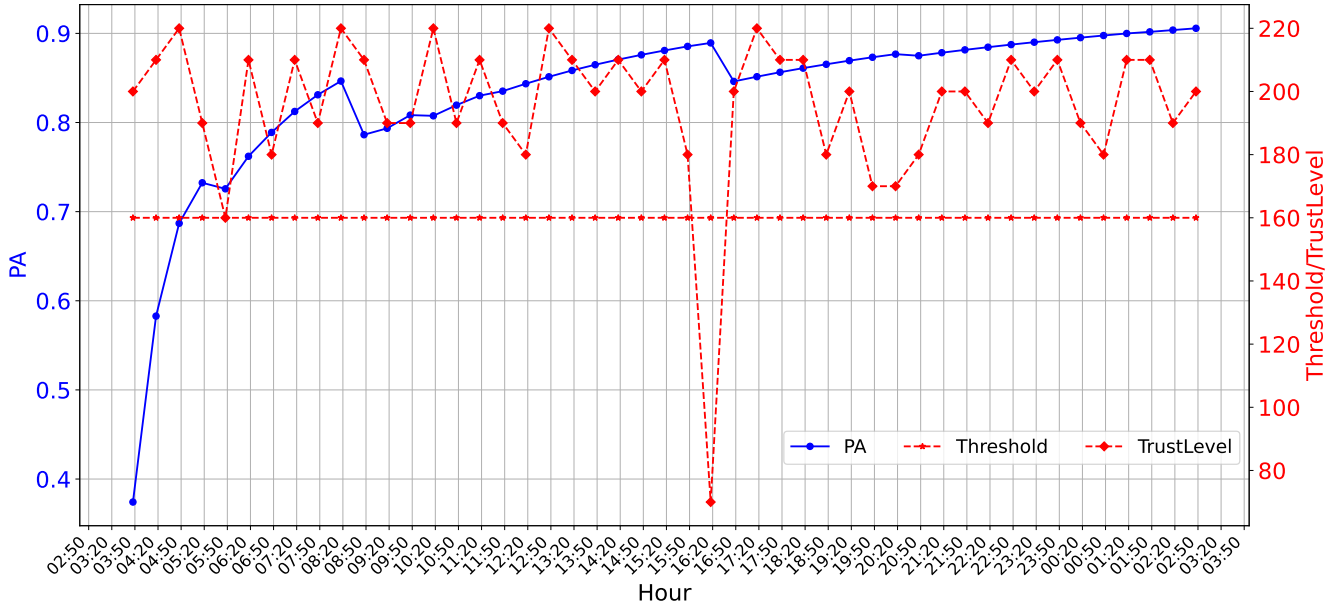

Figure 9: Network statistics in Strasbourg using Impact-FD.

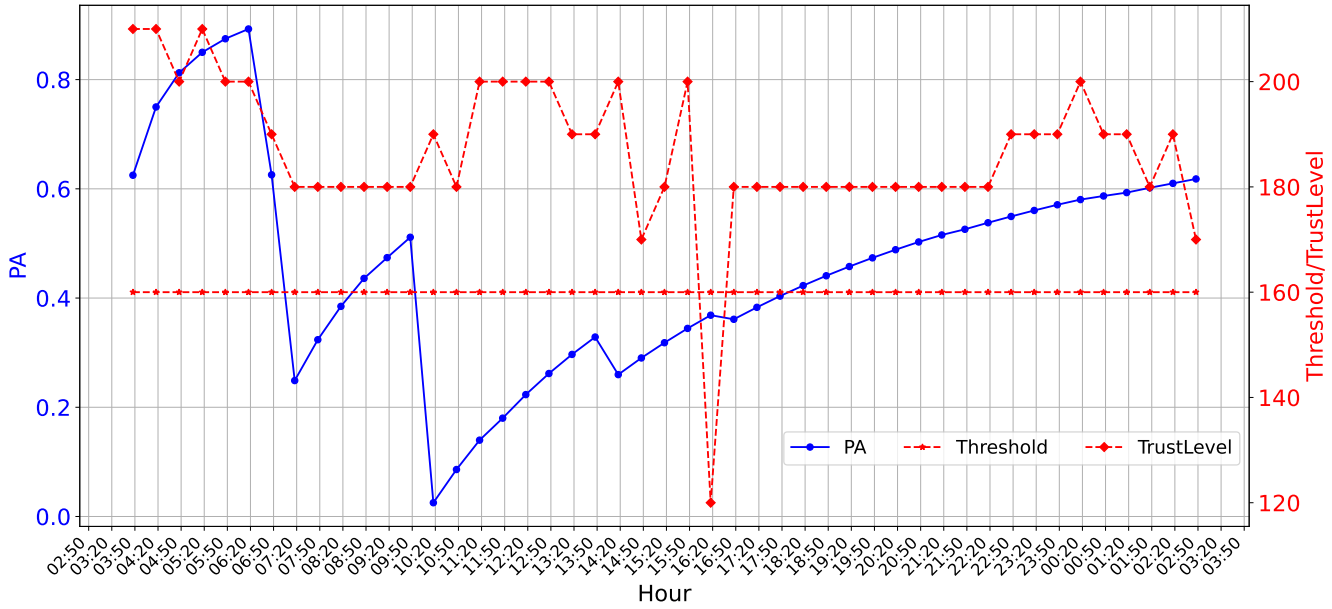

Figure 10: Network statistics in Grenoble using Impact-FD.

focusing on their methodologies, strengths, and limitations in the context of disaster-prone environments.

- **Impact-FD** [14] is a solution that proposes a new failure detector called Impact FD, which provides an output expressing the confidence of the failure detector about the system (or set of processes) as a whole. The confidence is configured by the impact factor, allowing the user to define the importance of each node within an acceptable margin of failure. Additionally, some flexibility properties are defined, which characterize the ability of Impact FD to tolerate a certain margin of failures or suspicions. The Disaster-FD extends Impact-FD, focusing on failure detection and network reliability assessment in IoT environments, with an emphasis on real-time monitoring. Additionally, Disaster-FD

performs federated monitoring across different regions and considers additional aspects such as device energy consumption.

- **Medley** [17] is a decentralized solution for failure detection in IoT operating in ad-hoc networks, employing spatial selection to send ping messages, and prioritizing nearby nodes. It uses multiple votes from various devices to assess a node's condition to reduce false positives. The SWIM protocol, integrated with Medley, ensures scalable and tolerant failure detection. On the other hand, Disaster-FD focuses on real-time monitoring and analysis of IoT networks, especially in disaster situations, prioritizing system reliability assessment.
- **Stab-FD** [15] is a solution that proposes a failure detector for distributed systems, especially suitable for wide area networks (WAN), which dynamically adjusts within a safety margin to adapt to variations in the quality of communication links. Additionally, Stab-FD has a cooperative version, where nodes exchange information about link stability and the list of suspected nodes [15]. Disaster-FD stands out in IoT environments, especially in disaster scenarios, using an algorithm to assess network reliability. Techniques proposed in Stab-FD for adaptation and reduction of false positives are orthogonal and could be applied to Disaster-FD.
- **SWIM**, the Scalable Weakly-consistent Infection-style Membership protocol, introduced by [9], is designed for failure detection in large-scale distributed systems. SWIM employs a combination of pinging and indirect probing to detect node failures efficiently. It is known for its scalability and robustness in large networks. While SWIM's principles of scalability are valuable, Disaster-FD integrates these concepts within the context of IoT networks, ensuring that the system remains efficient and reliable even in the face of large-scale disasters.

8 Conclusion

The presented study details the development and evaluation of Disaster-FD, a failure detector designed for disaster-prone environments, focusing on real-time monitoring of IoT networks. Inspired by Impact-FD [14], Disaster-FD introduces features such as federated monitoring and continuous network reliability assessment.

The results of tests conducted on the FIT-IoT-LAB platform demonstrate the effectiveness of Disaster-FD in detecting failures and monitoring network reliability in various scenarios, including communication instability situations. Disaster-FD's ability to maintain a stable trust level and high failure detection accuracy, even in unstable networks like Grenoble, highlights its robustness and applicability in real-world natural disaster scenarios.

Comparison with Impact-FD showed that Disaster-FD is more efficient in adapting to IoT network instabilities, resulting in less variability in trust levels and fewer false positives. Using the message identifier in estimating heartbeat arrival times contributes to the system's accuracy and stability. Therefore, Disaster-FD presents itself as a tool with the potential for monitoring IoT networks in disaster-prone environments, with greater resilience and consistency, essential for early failure detection and rapid response to adverse events.

Future work on Disaster-FD includes analyzing failure frequency to identify trends and determine the need for preventive maintenance; implementing failure-based notifications to anticipate potential natural disasters; applying the *ComputeMargin* function inspired by the Stab-FD project [15] to adjust monitoring timers based on communication link stability dynamically; and integrating with Telegram to send automatic alerts to system administrators in cases of network instability.

Acknowledgments

We thank the anonymous reviewers for their valuable feedback. This work was partially supported by the Brazilian Coordenação de Aperfeiçoamento de Pessoal de Nível Superior (CAPES) under grant 88881.368742/2019-01. Luciana Arantes and Pierre Sens acknowledge the support of the French Agence Nationale de la Recherche (ANR), under grant ANR-22-CE25-0008-01 (SkyData project).

References

[1] Cedric Adjih, Emmanuel Baccelli, Eric Fleury, Gaetan Harter, Nathalie Mitton, Thomas Noel, Roger Pissard-Gibollet, Frederic Saint-Marcel, Guillaume Schreiner, Julien Vandaele, et al. 2015. FIT IoT-LAB: A large scale open experimental IoT testbed. In *2015 IEEE 2nd World Forum on Internet of Things (WF-IoT)*. IEEE, 459–464.

[2] Marcos K Aguilera, Carole Delporte-Gallet, Hugues Fauconnier, and Sam Toueg. 2004. Communication-efficient leader election and consensus with limited link synchrony. In *Proceedings of the twenty-third annual ACM symposium on Principles of distributed computing*. 328–337.

[3] Luigi Atzori, Antonio Iera, and Giacomo Morabito. 2010. The internet of things: A survey. *Computer networks* 54, 15 (2010), 2787–2805.

[4] Michael Barborak, Anton Dahbura, and Miroslaw Malek. 1993. The consensus problem in fault-tolerant computing. *ACM Computing Surveys (CSur)* 25, 2 (1993), 171–220.

[5] Tushar Deepak Chandra, Vassos Hadzilacos, and Sam Toueg. 1996. The weakest failure detector for solving consensus. *Journal of the ACM (JACM)* 43, 4 (1996), 685–722.

[6] Tushar Deepak Chandra and Sam Toueg. 1996. Unreliable failure detectors for reliable distributed systems. *Journal of the ACM (JACM)* 43, 2 (1996), 225–267.

[7] Wei Chen, Sam Toueg, and Marcos Kawazoe Aguilera. 2002. On the quality of service of failure detectors. *IEEE Transactions on computers* 51, 5 (2002), 561–580.

[8] F. Cristian and C. Fetzer. 1999. The Timed Asynchronous Distributed System Model. *IEEE Transactions on Parallel and Distributed Systems* 10, 6 (1999), 642–657.

[9] Abhinandan Das, Indranil Gupta, and Ashish Motivala. 2002. Swim: Scalable weakly-consistent infection-style process group membership protocol. In *Proceedings International Conference on Dependable Systems and Networks*. IEEE, 303–312.

[10] Michael J Fischer, Nancy A Lynch, and Michael S Paterson. 1985. Impossibility of distributed consensus with one faulty process. *Journal of the ACM (JACM)* 32, 2 (1985), 374–382. https://dl.acm.org/doi/10.1145/3149.214121

[11] Lucas Janoneda. 2022. A cada desastre natural no Brasil, em média, 3,4 mil pessoas são afetadas. https://www.cnnbrasil.com.br/nacional/a-cada-desastre-natural-no-brasil-em-media-34-mil-pessoas-sao-afetadas Acessada: 2022-01-09.

[12] Matthias Kovatsch, Martin Lanter, and Zach Shelby. 2014. Californium: Scalable cloud services for the internet of things with coap. In *2014 International Conference on the Internet of Things (IOT)*. IEEE, 1–6.

[13] Lamport Leslie. 1998. The part-time parliament. *ACM Trans. on Computer Systems* 16 (1998), 133–169.

[14] Anubis Graciela de Moraes Rossetto, Cláudio FR Geyer, Luciana Arantes, and Pierre Sens. 2018. Impact fd: An unreliable failure detector based on process relevance and confidence in the system. *Comput. J.* 61, 10 (2018), 1557–1576.

[15] Pierre Sens, Luciana Arantes, Anubis Graciela De Moraes Rosseto, and Olivier Marin. 2024. Stab-FD: A Cooperative and Adaptive Failure Detector for Wide Area Networks. *J. Parallel and Distrib. Comput.* 186 (2024).

[16] P. Verissimo and L. Rodrigues. 2012. *Distributed Systems for System Architects*. Vol. 1. Springer Science & Business Media.

[17] Rui Yang, Shichu Zhu, Yifei Li, and Indranil Gupta. 2019. Medley: A novel distributed failure detector for IoT networks. In *Proceedings of the 20th International Middleware Conference*. 319–331. https://experts.illinois.edu/en/publications/medley-a-novel-distributed-failure-detector-for-iot-networks