# Trust-empowered, IoT-driven Legitimate Data Offloading

Zakaria Maamar
Zayed University
Dubai, UAE

Noura Faci
Claude Bernard Lyon 1 University
Lyon, France

Fadwa Yahya
Prince Sattam Bin Abdulaziz University
Al-Kharj, KSA
University of Sfax
Sfax, Tunisia

Ejub Kajan
State University of Novi Pazar
Novi Pazar, Serbia

## ABSTRACT

In an IoT environment deployed on top of fog and/or cloud nodes, offloading data between nodes is a common practice that aims at lessening the burden on these nodes and hence, meeting some real-time processing requirements. Existing initiatives put emphasis on “when to offload” and “where to offload” using criteria like resource constraint, load balancing, and data safety during transfer. However, there is limited emphasis on the trustworthiness of those nodes that will accept the offloaded data putting these data at risk of misuse. To address this limited emphasis, this paper advocates for trust as a decision criterion for identifying the appropriate nodes for hosting the offloaded data. A trust model is designed and then, developed considering factors like legitimacy, quality-of-service, and quality-of-experience. A system demonstrating the technical doability of the trust model is presented in the paper, as well.

## KEYWORDS

Data Flow, Internet-of-Things, Legitimacy, Offloading, Trust.

## 1 INTRODUCTION

On top of many forms of computing like enterprise computing, social computing, and ubiquitous computing in the field of Information and Communication Technologies (ICT), cloud computing and fog computing are among the latest ICT that organizations are embracing to tackle the challenges of the 21$^{st}$ century. On the one hand, cloud computing promotes Anything-as-a-Service (*aaS) as an operation-model for organizations that wish to concentrate on their core functionalities/competencies without being concerned with the availability of resources like software, platform, and infrastructure that would help them achieve these functionalities/competencies [16]. On the other hand, fog computing promotes the deployment of processing and/or storage resources at the edge of communication networks allowing to minimize data transfer and hence, exposure to interception risks [14].

A good amount of research discusses the synergy between cloud and fog [5, 19]. Indeed, despite cloud's pay-per-use and scalability benefits, cloud seems inappropriate for certain applications (e.g., medical and financial) that have strict non-functional requirements to satisfy in terms of minimizing data latency[1] and protecting sensitive data. Transferring data over public networks to (distant) clouds could take time because of high latency, could be subject to interceptions, alterations, and misuses, and could depend on network availability and reliability. To address data-latency and data-sensitivity concerns, ICT practitioners advocate for fog computing where processing and/or storage resources are made available “next” (or close) to where data is collected. According to Khebbeb et al. [10], cloud means *more* resources, *more* reliability, and *more* latency, and fog means *less* resources, *less* reliability, and *less* latency.

Although cloud and fog are expected to work hand-in-hand [12], there are situations where cloud and/or fog could seek the support of their respective peers with handling some complex cross-border business applications, for example. Known as offloading [13, 14], this support would lead to forming relations between $\mathcal{C}$louds ($\mathcal{C}$) [6], between $\mathcal{F}$ogs ($\mathcal{F}$) [2], and between $\mathcal{F}$ogs and $\mathcal{C}$louds. We refer to these 3 cases as $\mathcal{C}2\mathcal{C}$ offloading flow, $\mathcal{F}2\mathcal{F}$ offloading flow, and $\mathcal{F}2\mathcal{C}$ offloading flow. However, to ensure a successful offloading we advocate for some criteria with focus, in this paper, on trust and eligibility that would help identify the right peers according to past experiences, current loads, possible incentives, to cite just some. Questions that we raise, but not limited to, include how can clouds/fogs discover potential peers, what risks do clouds/fogs take when offloading demands to peers, what demands would be eligible for offloading, and what criteria would define clouds/fogs' trust and eligibility levels?

We adopt the Internet-of-Things (IoT) to illustrate trust-empowered legitimate offloading in an environment of multiple clouds and fogs. Millions of things (e.g., from tiny ones like chips to advanced ones like embedded systems) are spread over the cyber-physical environment collecting, processing, and distributing data of different types ranging from humidity level in a warehouse to number of vehicles on a highway. According to Gartner[2], 6.4 billion connected things were in use in 2016, up 3% from 2015, and will reach 20.8 billion by 2020. It is worth noting the strong coupling of IoT with cloud and fog computing as reported in the literature [5, 23]. Many IoT applications adopt cloud and fog as operation models to secure the necessary resources for processing, storing, and communicating the massive volume of data that things generate.

The rest of this paper is organized as follows. Section 2 briefly discusses cloud, fog, IoT, and trust. Section 3 presents the offloading model in terms of core concepts, types of data flows, and types of offloading flows. The way trust guides the definition of

[1] Puliafito et al. report that “*the average round trip time between an Amazon Cloud server in Virginia (U.S.A.) and a device in the U.S. Pacific Coast is 66ms; it is equal to 125ms if the end device is in Italy; and reaches 302ms when the device is in Beijing*” [17].

[2] www.gartner.com/newsroom/id/3165317.

offloading flows is presented in the same section. Section 4 discusses the technical details about the offloading model. Section 5 concludes the paper and presents some future work.

## 2 BACKGROUND

Cloud is a popular ICT topic that promotes Anything-as-a-Service operation model, adopts pay-per-use pricing, and consolidates hardware and software resources into datacenters. Cloud computing is sometimes known as the $5^{th}$ utility after water, electricity, gas, and telephony. However, despite cloud popularity, it does not, unfortunately, suit all applications. It is not recommended for latency-critical and data-sensitive applications due to reasons such as high latency added by network connections to datacenters and multi-hops/nodes between end-users and datacenters that increase the probability of interceptions.

Fog was generalized by Cisco Systems in 2014 [4] as a new ICT-based operation model. The main idea is to make processing, storage, and networking resources "close" to data. Real-time applications that require almost immediate action and high data protection, would discard cloud in favor of fog. Varghese et al. mention that by 2020, existing electronic devices will generate 43 trillion gigabytes of data that need to be processed in cloud datacenters [22]. However, this way of operating cannot be sustained for a long time due to frequency and latency of communication between these devices and cloud datacenters. Fog would process data closer to its source so, that, network traffic is reduced and both Quality-of-Service (QoS) and Quality-of-Experience (QoE) are improved.

The abundant literature about IoT does not help propose a unique definition for IoT. We present 3 references on IoT that are [3], [1], and [18]. First, Barnaghi and Sheth provide a good overview of IoT requirements and challenges [3]. On the one hand, requirements include quality, latency, trust, availability, reliability, and continuity that should impact efficient access and use of IoT data and services. On the other hand, challenges result from today's IoT ecosystems that feature billions of dynamic things that make existing search, discovery, and access techniques and solutions inappropriate for IoT data and services. Second, Abdmeziem et al. discuss IoT characteristics and enabling technologies [1]. Characteristics include distribution, interoperability, scalability, resource scarcity, and security. And, enabling technologies include sensing, communication, and actuating. These technologies are mapped onto a three-layer IoT architecture that are referred to as perception, network, and application, respectively. Finally, Qin et al. [18] define IoT from a data perspective as *"In the context of the Internet, addressable and interconnected things, instead of humans, act as the main data producers, as well as the main data consumers. Computers will be able to learn and gain information and knowledge to solve real world problems directly with the data fed from things. As an ultimate goal, computers enabled by the Internet of Things technologies will be able to sense and react to the real world for humans".*

Trust may be seen as "*an act of faith; confidence and reliance in something that's expected to behave or deliver as promised*" [9]. On a regular basis, many ICT researchers and practitioners raise the question of should we trust cloud services or not. In [8], Huang and Nicol examine reputation- and Service-Level Agreement (SLA)-based trust, and Trust-as-a-Service (TaaS). While reputation-based trust relies on a user's (or community of users) experience, SLA-based trust relies on QoS measurement and SLA verification, in our case between clouds. The former may be weak in term of transparency since a cloud service's provider may fine tune/beef up some measurements. Regarding TaaS, trust management is delegated to a third party. However, some policy and security mechanisms should be considered, like accreditation and Public Key Infrastructure (PKI), and should also include a cloud auditor in charge of certifying both the cloud provider and the third party. In [9], Khan and Malluhi promote first, cloud prevention to ensure data privacy and access control and second, digital signature to ensure data integrity. The authors raise the question of "*how can cloud providers earn their customers' trust when a third party is processing sensitive data in a remote machine located in various countries?*". In [11], Li presents FASTCloud that is a framework to assess and select trustworthy cloud service based on QoS. This framework's 4 main components are cloud service providers, cloud service customers, potential cloud consumers, and a trustworthy cloud service selection. The last one evaluates trust level of cloud services based on the collected QoS attributes information by employing the trust assessment model, and returning the trust assessment results to potential cloud consumers.

## 3 OFFLOADING MODEL

This section discusses data flows in conjunction with offloading flows and then, presents the impact of trust on offloading decisions.

### 3.1 Overview

According to Mahmud et al., offloading in the context of fog-based applications could be bottom-up, top-down, and hybrid [14]. Using Fig. 1, we illustrate the cases of offloading flows (of), $\mathcal{C}2\mathcal{C}_{\text{of}}$, $\mathcal{F}2\mathcal{F}_{\text{of}}$, and $\mathcal{F}2\mathcal{C}_{\text{of}}$, that could arise in conjunction with data flows (df) that we specialize into $\mathcal{T}2\mathcal{C}_{\text{df}}$, $\mathcal{T}2\mathcal{F}_{\text{df}}$, and $\mathcal{F}2\mathcal{C}_{\text{df}}$ where $\mathcal{T}$ refers to $\mathcal{T}$hing. df is meant for capturing data that things collect/sense and conveying these data to recipients namely, cloud and fog, depending on the under-development IoT applications' functional and non-functional requirements [13]. Conveyed data could be either raw or processed depending again on the recipients' needs and these requirements, as well.

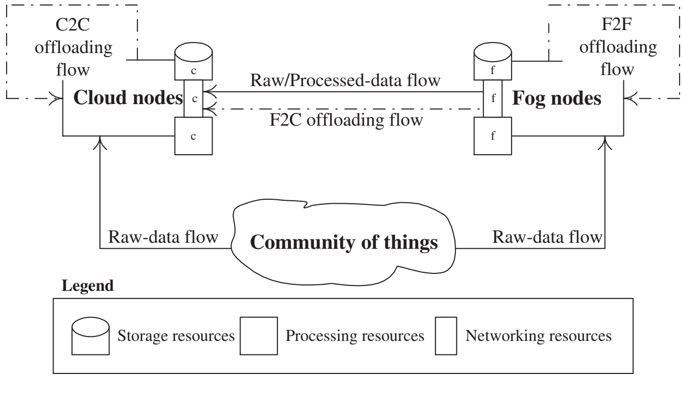

**Figure 1: Offloading flows in conjunction with data flows**

### 3.2 Types of data flows

To define data flows, we rely on our previous work on cloud-fog coordination [12, 24]. As stated above, data flows connect thing and fog together ($\mathcal{T}2\mathcal{F}_{\text{df}}$), thing and cloud together ($\mathcal{T}2\mathcal{C}_{\text{df}}$), and fog and cloud together ($\mathcal{F}2C_{\text{df}}$)[3]. Although the 3 specialized data flows could simultaneously exist, we came up in our previous work with 6 criteria whose use would permit to Highly-Recommend (HR), Recommend (R), and Not-Recommend (NR) which data flow should exist for an under-development IoT application. These criteria are *frequency* (rate of data transfer from things to fogs/clouds; the frequency could be regular, e.g., every 2 hours, or continuous), *sensitivity* (nature of data exchanged between things and fogs/clouds; highly-sensitive data should not be exposed longer on networks during the exchange), *freshness* (how important data exchanged between things and fogs/clouds should be up-to-date, i.e., recent), *time* (delay that results from withholding/processing data at the thing level until its transfer to fogs/clouds), *volume* (amount of data that things produce and send to fogs/clouds), and *criticality* (demands that fogs/clouds express with regard to data of things; low demands could lead to ignoring certain data). Assumption made in support of the 6 criteria is that, distance-wise, clouds are **far** from things and fogs are **close** to things.

In Table 1, we summarize how the afore-mentioned criteria, taken independently from each other, assist with recommending the establishment of specific data flows. More details about these recommendations are presented in [12].

Contrarily to what we did in [12] where *frequency*, *sensitivity*, *freshness*, *time*, *volume*, and *criticality* are taken independently from each other, we combined them all using a fuzzy logic-based multi-criteria decision making approach [24]. This approach was demonstrated using a healthcare-driven IoT application along with an in-house testbed that featured real sensors (temperature and humidity DHT11) and fog (rPi2) and cloud (Ubidots) platforms. During the experiments, we modified the *frequency* of streaming data (every 3 second, 5 second, 7 second, and randomly) for each of the 3 data flows, $\mathcal{T}2C$, $\mathcal{T}2\mathcal{F}$, and $\mathcal{F}2C$, and the *volume* (around low and high amount) and *criticality* (around low and high important) of the transmitted data. Upon data receipt at an end-point whether fog or cloud, we timestamped data messages prior to storing them. Table 3 summarizes the experiments with focus on the recommendations of establishing specific data flows. More details about these recommendations are presented in [24].

## 3.3 Types of offloading flows

$C2C_{\text{of}}$, $\mathcal{F}2\mathcal{F}_{\text{of}}$, and $\mathcal{F}2C_{\text{of}}$ identify possible interactions between clouds, between fogs, and between fogs and clouds. As stated earlier, the objective of offloading is to secure the support of all parties that could either be "idle" or have a "light" load allowing them to accommodate additional demands from peers. It is worth noting that offloading flows are to a certain extent in-line with data flows shedding light on the loads that clouds and fogs could handle depending on the volume of data that things would submit for processing and/or storage.

(1) $C2C_{\text{of}}$ establishes collaboration between clouds according to their ongoing loads and processing and storage resources. In compliance with Fig. 1, things periodically collect and generate (raw) data from the cyber-physical surroundings and send these data to clouds ($\mathcal{T}2C_{\text{df}}$) and/or fogs ($\mathcal{T}2\mathcal{F}_{\text{df}}$) for processing/storage, as deemed necessary (Section 3.2). A cloud can serve a certain number of data-based demands instantly or offload some to other reachable clouds in the same domain if this cloud is congested, which could delay handling these demands. While the offloading could be based on peers' current loads (and other performance criteria like storage capacity), we examine in Section 3.4 the value-added of trust in selecting these peers.

(2) $\mathcal{F}2\mathcal{F}_{\text{of}}$ establishes collaboration between fogs according to their ongoing loads and processing and storage resources [2]. In compliance with Fig. 1, things periodically collect and generate (raw) data from the cyber-physical surroundings and send these data to fogs ($\mathcal{T}2\mathcal{F}_{\text{df}}$) and/or clouds ($\mathcal{T}2C_{\text{df}}$) for processing/storage, as deemed necessary (Section 3.2). A fog can serve a certain number of data-based demands instantly or offload some to other reachable fogs in the same domain if this fog is congested, which could delay handling these demands. While the offloading could be based on peers' current loads (and other performance criteria like storage capacity), we examine in Section 3.4 the value-added of trust in selecting these peers.

(3) $\mathcal{F}2C_{\text{of}}$ establishes collaboration between fogs and clouds when these fogs' offloading demands cannot be accommodated by other fogs in the context of $\mathcal{F}2\mathcal{F}_{\text{of}}$. Performance and/or trust criteria could back the decision of discarding these fogs. In compliance with Fig. 1, fogs could send (either raw or processed) data to clouds ($\mathcal{F}2C_{\text{df}}$) for (extra) processing/storage, as deemed necessary. While the offloading could be based on clouds' current loads (and other performance criteria like storage capacity), we examine in Section 3.4 the value-added of trust in selecting these clouds.

## 3.4 Trust-empowered legitimate offloading

Table 3 contains the list of parameters used to calculate $\mathcal{T}$rust $\mathcal{S}$cores ($\mathcal{TS}$) of fogs and clouds. Among these parameters, we cite list of acquaintances, QoS, and QoE.

In [7], Fiedler et al. suggest that different factors could influence $QoE$. In our work, we adopt one influence factor that is $QoS$ allowing to compute $QoE$ as per Equation 1.

$$QoE = \begin{cases} good & \text{if } |QoS^A - QoS^M| \in [\sigma + \delta, 1] \\ fair & \text{if } |QoS^A - QoS^M| \in [\sigma - \delta, \sigma + \delta[ \\ bad & \text{if } |QoS^A - QoS^M| \in [0, \sigma - \delta[ \end{cases} \qquad (1)$$

where ($\sigma \pm \delta$) would define a threshold.

$\mathcal{TS}$ calculation begins with identifying potential connections between data flows, between offloading flows, and between data flows and offloading flows. The objective of this identification is to determine who initiates what; a flow's recipient could initiate another flow and so on. In Fig. 2, dashed lines correspond to these connections that we label as "could be the same". In the context of trust-empowered legitimate offloading, 4 out of 5 "could be the same" connections constitute our focus as per the following cases (only 2 are detailed):

**Case 1.** $\mathcal{T}^i2\mathcal{F}^j_{\text{df}} \longrightarrow \mathcal{F}^j2\mathcal{F}^k_{\text{of}}|C^k_{\text{of}}$: following the formation of a data flow from $\mathcal{T}^i$ to $\mathcal{F}^j$, $\mathcal{F}^j$ assesses its current processing and storage resources and, then, decides to form an offloading load to convey the received (raw) data to $\mathcal{F}^{k \neq j}$, $C^k$, or both $\mathcal{F}^{k \neq j}$ and $C^k$. Detecting $\mathcal{F}^{k \neq j}$ and $C^k$ is made possible thanks to acq($\mathcal{F}^j$). When calculating the trust scores of $\mathcal{F}^k$ and/or $C^k$ and since $\mathcal{F}^j$ would financially compensate $\mathcal{F}^k$ and/or $C^k$, we adopt a conservative

[3] Pre-processing data at fogs prior to sending the pre-processed data to clouds.

Table 1: Recommendations about establishing data flows when criteria are separated ([12])

| Criterion | Features | $\mathcal{T}2\mathcal{C}_{\text{df}}$ | $\mathcal{T}2\mathcal{F}_{\text{df}}$ | $\mathcal{F}2\mathcal{C}_{\text{df}}$ |
|---|---|---|---|---|
| *Frequency* | Continuous stream | NR | HR | R |
| | Regular stream<br>Short gaps<br>Long gaps | <br>NR<br>R | <br>HR<br>R | <br>HR<br>R |
| *Sensitivity* | High | NR | HR | HR |
| | Low | R | R | R |
| *Freshness* | Highly important | NR | HR | R |
| | Lowly important | R | R | R |
| *Time* | Real-time | NR | HR | HR |
| | Near real-time | R | HR | HR |
| | Batch-processing | HR | NR | NR |
| *Volume* | High amount | HR | NR | NR |
| | Low amount | NR | HR | R |
| *Criticality* | Highly important | HR | HR | R |
| | Lowly important | NR | HR | HR |

Table 2: Recommendations about establishing data flows when criteria are combined ([24])

| Scenario # | Criteria | Linguistic values | Recommendations |
|---|---|---|---|
| Scenario 1 | *Frequency* | Regular stream (around short and long gaps)$^\star$ | $\mathcal{T}2\mathcal{C}_{\text{df}}$ is NR; $\mathcal{T}2\mathcal{F}_{\text{df}}$ is R; $\mathcal{F}2\mathcal{C}_{\text{df}}$ is R |
| | *Sensitivity* | Around low and high$^\star$ | |
| | *Freshness* | Highly important | |
| | *Time* | Real time | |
| | *Volume* | High amount | |
| | *Criticality* | Lowly important | |
| Scenario 2 | *Frequency* | Regular stream long gaps | $\mathcal{T}2\mathcal{C}_{\text{df}}$ is NR; $\mathcal{T}2\mathcal{F}_{\text{df}}$ is HR; $\mathcal{F}2\mathcal{C}_{\text{df}}$ is R |
| | *Sensitivity* | High | |
| | *Freshness* | Highly important | |
| | *Time* | Real time | |
| | *Volume* | Low amount | |
| | *Criticality* | Lowly important | |
| Scenario 3 | *Frequency* | Regular stream long gaps | $\mathcal{T}2\mathcal{C}_{\text{df}}$ is R; $\mathcal{T}2\mathcal{F}_{\text{df}}$ is R; $\mathcal{F}2\mathcal{C}_{\text{df}}$ is R |
| | Sensitivity | Low | |
| | Freshness | Lowly important | |
| | Time | Near-real time | |
| | Volume | Around low and high amount$^\star$ | |
| | Criticality | Highly important | |
| Scenario 4 | *Frequency* | Regular stream long gaps | $\mathcal{T}2\mathcal{C}_{\text{df}}$ is R; $\mathcal{T}2\mathcal{F}_{\text{df}}$ is R; $\mathcal{F}2\mathcal{C}_{\text{df}}$ is R |
| | *Sensitivity* | Low | |
| | *Freshness* | Lowly important | |
| | *Time* | Near-real time | |
| | *Volume* | High amount | |
| | *Criticality* | Around lowly and highly important$^\star$ | |

$^\star$: Around Val$_1$ and Val$_2$: both Val$_1$ and Val$_2$ meet the scenario's requirements.

Table 3: List of parameters

| Parameter | Description |
|---|---|
| $\mathcal{C}$ | Set of all clouds in the ecosystem. |
| $\mathcal{F}$ | Set of all fogs in the ecosystem. |
| $\mathcal{T}$ | Set of all things in the ecosystem. |
| $acq(entity_i)$ | Acquaintance function that returns entities (things, fogs, and/or clouds) within range of entity$_i$. |
| $QoS^A_{\mathcal{C}^i2\mathcal{C}^j}$ | Announced Quality of Service that $\mathcal{C}^j$ is expected to maintain when accepting $\mathcal{C}^i$'s offloading demands. Announced $QoS$ is compared to Measured $QoS$ ($QoS^M_{\mathcal{C}^i2\mathcal{C}^j}$) to detect any gap (Equation 1). |
| $QoS^A_{\mathcal{F}^i2\mathcal{F}^j}$ | Announced Quality of Service that $\mathcal{F}^j$ is expected to maintain when accepting $\mathcal{F}^i$'s offloading demands. Announced $QoS$ is compared to Measured $QoS$ ($QoS^M_{\mathcal{F}^i2\mathcal{F}^j}$) to detect any gap (Equation 1). |
| $QoS^A_{\mathcal{F}^i2\mathcal{C}^j}$ | Announced Quality of Service that $\mathcal{C}^j$ is expected to maintain when accepting $\mathcal{F}^i$'s offloading demands. Announced $QoS$ is compared to Measured $QoS$ ($QoS^M_{\mathcal{F}^i2\mathcal{C}^j}$) to detect any gap (Equation 1). |
| $QoE_{\mathcal{C}^i2\mathcal{C}^j}$ | Quality of Experience that $\mathcal{C}^i$ uses to capture its satisfaction in $\mathcal{C}^j$ completing its offloading demands. $QoE$ is dependent on whether $\mathcal{C}^j$'s $QoS$ was maintained or not at run-time[4]. |
| $QoE_{\mathcal{F}^i2\mathcal{F}^j}$ | Quality of Experience that $\mathcal{F}^i$ uses to capture its satisfaction in $\mathcal{F}^j$ completing its offloading demands. $QoE$ is dependent on whether $\mathcal{F}^j$'s $QoS$ was maintained or not at run-time. |
| $QoE_{\mathcal{F}^i2\mathcal{C}^j}$ | Quality of Experience that $\mathcal{F}^i$ uses to capture its satisfaction in $\mathcal{C}^j$ completing its offloading demands. $QoE$ is dependent on whether $\mathcal{C}^j$'s $QoS$ was maintained or not at run-time. |
| $n_{\mathcal{C}^i2\mathcal{C}^j}$ | Number of times that $\mathcal{C}^j$ accepted/completed the offloading demands of $\mathcal{C}^i$. |
| $n_{\mathcal{F}^i2\mathcal{F}^j}$ | Number of times that $\mathcal{F}^j$ accepted/completed the offloading demands of $\mathcal{F}^i$. |
| $n_{\mathcal{F}^i2\mathcal{C}^j}$ | Number of times that $\mathcal{C}^j$ accepted/completed the offloading demands of $\mathcal{F}^i$. |

approach that consists of making $\mathcal{F}^j$ check the authenticity of the data's sender, namely $\mathcal{T}^i$, using $\mathcal{T}^i2\mathcal{F}^j_{\text{df}}$. To this end, we relate authenticity to a flow's initiator whether this initiator would be a thing, a fog, or a cloud and define the concept of legitimate initiator [25]. Here, $\mathcal{T}^i$ is the initiator and its legitimacy becomes a concern for $\mathcal{F}^j$.

According to Pakulski, "*legitimacy therefore relies not on trust, but on an impersonal sense of duty on the part of the followers to follow commands of a proper authority, whoever is in authority, and whatever is the content of these commands*" [15].

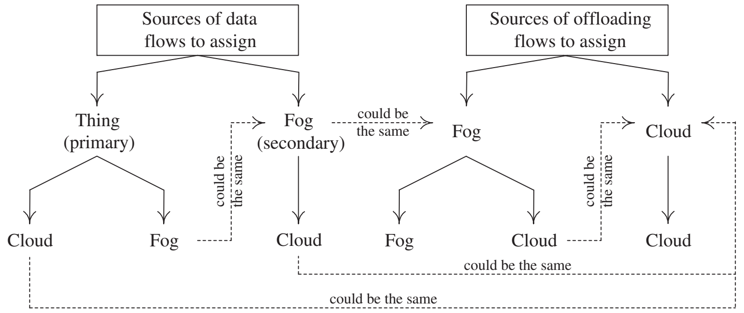

**Figure 2: Potential connections between the different flows**

For illustration, let us assume that $\mathcal{T}^i$ is illegitimate by flooding $\mathcal{F}^j$ with "fake" data, which triggers the formation of an offloading flow from $\mathcal{F}^j$ to $\mathcal{F}^k$. Processing these data at the level of $\mathcal{F}^k$ produces irrelevant results for users along with wasting $\mathcal{F}^j$'s financial resources and $\mathcal{F}^k$'s processing resources as well as "blaming" $\mathcal{F}^k$ for these results. At the end, $\mathcal{F}^j$ adjusts the trust score of $\mathcal{F}^k$. To avoid this scenario, our approach to calculate trust scores considers the legitimacy of a flow's initiator along with the quality of experience that results from handling this flow. Due to legitimacy aspect, we refine $QoE_{\mathcal{F}^j 2\mathcal{F}^k}$ into $QoE^{\mathcal{T}^i}_{\mathcal{F}^j 2\mathcal{F}^k}$ where $\mathcal{T}^i$ is the reason of making $\mathcal{F}^j$ interact with $\mathcal{F}^k$. We compute a data flow-triggered $\mathcal{TS}^{\text{df}}_{\mathcal{F}^j,\mathcal{F}^k}$ using Equation 2.

$$\mathcal{TS}^{\text{df}}_{\mathcal{F}^j,\mathcal{F}^k} = \text{Agg}(\{Leg_{\mathcal{T}^i,\mathcal{F}^j} \times QoE^{\mathcal{T}^i}_{\mathcal{F}^j 2\mathcal{F}^k}\}_{i=1,n}) \tag{2}$$

where

- Agg refers to some common aggregate function like average and minimum.
- $Leg_{\mathcal{T}^i,\mathcal{F}^j}$ is $\mathcal{T}^i$'s legitimacy when establishing a data flow with $\mathcal{F}^j$.

To define $\mathcal{T}^i$'s legitimacy, we develop behavioral patterns like those presented in [20]. The objective is to demystify the recurrent behavior of $\mathcal{T}^i$ based on its interactions with fogs and/or clouds. We link behavioral patterns to the recommendations presented in Table 2 and specialize them into legitimate pattern ($\mathcal{L}_p$), where a thing would implement HR and R outcomes that were obtained with respect to IoT applications' features/linguistic values (e.g., real-time and continuous streaming), and illegitimate pattern ($\mathcal{I}_p$), where a thing would do the opposite by implementing NR outcomes. We compute $Leg_{\mathcal{T}^i,\mathcal{F}^j}$ using Equation 3.

$$Leg_{\mathcal{T}^i,\mathcal{F}^j} = \mathcal{V}(\text{App}_{\mathcal{T}^i,\mathcal{F}^j}, \mathcal{T}^i 2\mathcal{F}^j_{\text{df}}) \tag{3}$$

where

- $\text{App}_{\mathcal{T}^i,\mathcal{F}^j}$ is the IoT application in which $\mathcal{T}^i$ and $\mathcal{F}^j$ jointly participate.
- $\mathcal{V}(\text{App}_{\mathcal{T}^i,\mathcal{F}^j}, \mathcal{T}^i 2\mathcal{F}^j_{\text{df}})$ refers to $\mathcal{T}^i$'s legitimacy $\mathcal{V}$alue to send data to $\mathcal{F}^j$ during $\text{App}_{\mathcal{T}^i,,\mathcal{F}^j}$'s execution.

$$\mathcal{V}(\text{App}_{\mathcal{T}^i,\mathcal{F}^j}, \mathcal{T}^i 2\mathcal{F}^j_{\text{df}}) = \begin{cases} 1 & \because \text{the data flow is highly recommended.} \\ 0 & \because \text{the data flow is recommended.} \\ \text{-1} & \because \text{the data flow is not recommended.} \end{cases}$$

**Case 2.** $\mathcal{F}^i 2C^j_{\text{of}} \longrightarrow C^j 2C^k_{\text{of}}$: following the formation of an offloading flow (that most probably would be connected to a data flow from $\mathcal{F}^i$ to $C^j$, $C^j$ assesses its current processing and storage resources and, then, decides to convey this offloading flow to $C^{k \neq j}$. When $C^j$ calculates the trust score of $C^k$, we deem necessary to include the trust score, that $\mathcal{F}^i$ would have defined for $C^j$, in this calculation. The rationale of this inclusion is that $\mathcal{F}^i$ would like to "know" to whom its offloading flow would be assigned since its initial contact for the offloading is $C^j$ and not $C^k$. Compared to case 1, $\mathcal{F}^i$'s eligibility is not a concern based on the previous trust score calculation that involved $\mathcal{F}^i$ and $C^j$. After refining $QoE_{C^j 2C^k}$ into $QoE^{\mathcal{F}^i}_{C^j 2C^k}$ where $\mathcal{F}^i$ is the reason of making $C^j$ interact with $C^k$, we compute an offloading flow-triggered $\mathcal{TS}^{\text{of}}_{C^j,C^k}$ using Equation 4.

$$\mathcal{TS}^{\text{of}}_{C^j,C^k} = \text{Agg}(\{\mathcal{TS}^{\text{of}}_{\mathcal{F}^i,C^j} \times QoE^{\mathcal{F}^i}_{C^j 2C^k}\}_{i=1,n}) \tag{4}$$

where, compared to Equation 2's $\mathcal{TS}^{\text{df}}$, Equation 4's $\mathcal{TS}^{\text{of}}$ refers to trust-score calculation that is triggered because of an offloading flow and not data flow and is defined as follows:

$$\mathcal{TS}^{\text{of}}_{\mathcal{F}^i,C^j} = \text{Agg}(\{QoE^{C^p}_{\mathcal{F}^i 2C^j}\}_{C^p \in \mathcal{X}_{i,j}}) \times e^{-|\mathcal{X}_{i,j}|} \tag{5}$$

where

- $\mathcal{X}_{i,j}$ denotes the multiset of peers (including $C^k$) to which $C^j$ forwarded the offloading demands of $F^i$.
- $QoE^{C^p}_{\mathcal{F}^i 2C^j}$ indicates $F^i$'s offloading quality-of-experience with $C^j$ given $C^p$ in $\mathcal{X}_{i,j}$.
- $|\mathcal{X}_{i,j}|$ represents $\mathcal{X}_{i,j}$'s cardinality.

**Case 3.** $\mathcal{T}^i 2C^j_{\text{df}} \longrightarrow C^j 2C^k_{\text{of}}$ following the formation of a data flow from $\mathcal{T}^i$ to $C^j$, $C^j$ assesses its current processing and storage resources and then, decides to form an offloading load to convey the received (raw) data to $C^{k \neq j}$.

**Case 4.** $\mathcal{F}^i 2C^j_{\text{df}} \longrightarrow C^j 2C^k_{\text{of}}$: following the formation of a data flow from $\mathcal{F}^i$ to $C^j$, $C^j$ assesses its current processing and storage resources and then, decides to form an offloading load to convey the received (processed) data to $C^{k \neq j}$, for extra-processing.

## 4 EXPERIMENTS

This section discusses the testbed and experiments to validate the offloading model and then, presents some results.

## 4.1 Testbed set-up

To check the technical doability of our offloading model, we deployed a testbed that simulates both data flows between things and clouds/fogs and offloading flows between fogs, between clouds, and between fogs and clouds. Through the testbed, we aimed at selecting the trustworthy recipient of an offloading flow. The testbed is fully developed in Java SE 8 under Eclipse IDE for Java Developers[5]. The computer used during development runs Windows 8 64 bits, 1.4 GHz quad-core processor CPU, and 4GB RAM. The development focused on 3 decision-making components discussed below:

- *Thing decider* runs over thing nodes to identify where data of these things should be sent. This decider refers to our cloud-fog coordination work (Table 2) and determines the legitimacy of things when selecting data recipients as per Equation 3. For instance, if fog is highly-recommended and cloud is not-recommended as per Table 2, then the *thing decider* will privilege fog nodes over cloud nodes. Should the *thing decider* comply with this recommendation, then it assigns 1 as a legitimacy value to the thing sending data to one of the fog to select.
- *Fog offloading decider* runs over fog nodes to identify a trustworthy recipient of data when offloading data becomes necessary. Indeed, each fog in the testbed has its *fog offloading decider* that accesses details stored in an in-house developed database about past experiences between various nodes (things, fogs, and clouds) of the testbed. Such details include legitimacy of thing nodes and fog/cloud nodes' announced and measured QoSs. It is worth mentioning that we used a QoS dataset[6] to assign QoS values to fog and cloud nodes when data flows or offloading flows are required. Based on thing's legitimacy and cloud/fog QoSs, the *fog offloading decider* computes $\mathcal{TS}$ for all acquaintances of a respective fog node. Finally, it selects the node with the highest $\mathcal{TS}$.
- *Cloud offloading decider* runs on top of cloud nodes acting like *fog offloading decider.* Contrarily to fogs, clouds can offload data to their peers, only.

## 4.2 Simulations and results

To validate both the offloading model and the decision makers' outcomes, we carried out different experiments. First, an in-house client-server Java application, $Tapp$, allowed to simulate things' behaviors as clients sending data extracted from a *T-drive Taxi Trajectories* dataset [26] to clouds and fogs that act as servers. We annotated this dataset with data-flow recommendations (Table 2) so that, the *thing decider* selects the best data flow recipient. Second, we developed another client-server Java application, $CFapp$, to simulate clouds' and fogs' behaviors when offloading data. A cloud/fog is simultaneously a client when it offloads data and a server when it receives data. Each cloud/fog's $CFapp$ receives data sent by other nodes, stores them, and decides to offload them to other nodes. As per Table 4, both Java applications run on various types of computers that vary between personal laptops located in KSA and Tunisia and desktops available in the laboratories of Prince Sattam Bin Abdulaziz University in KSA.

To perform the experiments, we simulated data flows from things to clouds/fogs. A data flow contains raw data in terms of id, message extracted from *T-drive Taxi Trajectories* dataset, and sending timestamps. Here, the *thing decider* coupled to $Tapp$ selects the adequate recipient of data flows that could be a cloud or a fog (Table 2). In addition, the *thing decider* computes and stores the legitimacy of things per data flow in a database. We associate the experiments with the following scenarios:

- *Scenario #1* simulates the offloading flows from a fog to fogs. These flows are triggered by data flows received from things (case 1, Section 3.4). To this end, the *fog offloading decider* coupled to $CFapp$ selects the adequate recipient of the offloading flows. Indeed, the decider computes the trust score (Equation 2) for all acquaintances of the current fog node prior to selecting the one with the highest trust score.
- *Scenario #2* is built upon *scenario #1* where simulations are about offloading flows triggered by data flows but expected to be offloaded from a fog to a cloud (case 1, Section 3.4). Here, the *fog offloading decider* selects the adequate cloud recipient of the offloading flows based on the calculated trust scores (Equation 2).
- *Scenario #3* simulates offloading flows triggered by previous offloading flows from a cloud node another cloud node (case 2, Section 3.4). Here, the *cloud offloading decider* selects the best recipient of the offloading flows based on the calculated trust scores (Equation4).

In conjunction with the experiments, we examined the variations in selecting data recipients depending on who offloads data, either cloud or fog, so, that, we highlight the impact of trust-empowered legitimate offloading on the selection of data recipient. Deployed on many cloud/fog nodes, $CFapp$ could decide to offload the received data to another node in the network. The *cloud/fog offloading decider* selects the recipient based on their trust scores. Fig. 3 to Fig. 5 illustrate the number of offloading flows received by each node in the network with focus on the variation of this number over time. Indeed, the number of offloading flows received by a node increases when the node maintains its trust score and decreased due a degradation of its trust score. The variation of trust scores is further detailed in Fig. 6 and Fig. 7 measuring the trust scores of some nodes during the experiments. In addition, these figures focus on the impact of the variation in calculated trust scores on the selected node to receive the offloading flows. As per Fig. 3, during scenario #1 the 3 fogs, $F1$, $F2$, and $F3$, are selected as recipient of offloaded data. The experiments prove that the selection could vary according to the trust score that is in turn calculated based on the QoE and legitimacy of things. For instance, $F2$, which was privileged at the beginning of the experiments, is penalized after a degradation of its trust score. Fig. 6 shows only the trust scores calculated by $F1$. This figure shows that $C1$ having the highest trust score at the beginning of the experiments was selected by $F1$ as offloading recipient. Then, and due a degradation in $C1$'s trust score, $C2$ having the new highest trust score is selected by $F1$ as offloading recipient.

# 5 CONCLUSION

In an IoT environment consisting of multiple fogs and clouds, it happens that data that things send, whether separately or concurrently, to these fogs and/or clouds for processing and/or storage needs end-up being transferred to other peers for the same needs. Known as offloading, this paper stresses out the importance of ensuring the trustworthiness of data recipients to avoid

[5] www.eclipse.org/downloads/packages.

[6] github.com/QXL4515/data-set.

**Table 4: List of used computers**

| Computer | Connection | Location | Role in the testbed |
|---|---|---|---|
| Laptop 1 | WiFi | KSA | Fog |
| Laptop 2 | WiFi | KSA | Cloud |
| Laptop 3 | WiFi | Tunisia | Cloud |
| Laptop 4 | WiFi | Tunisia | Cloud |
| Desktop x 6 | Ethernet | KSA | 2 Fogs and 4 Things |

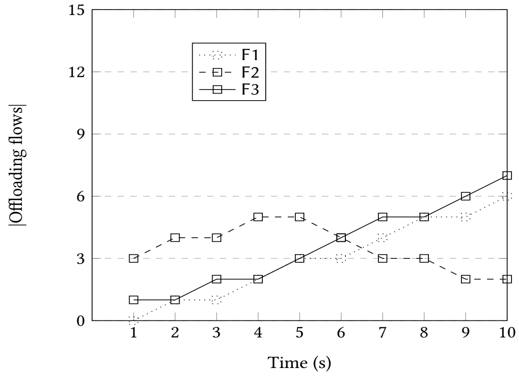

**Figure 3: Results of scenario #1 experiments**

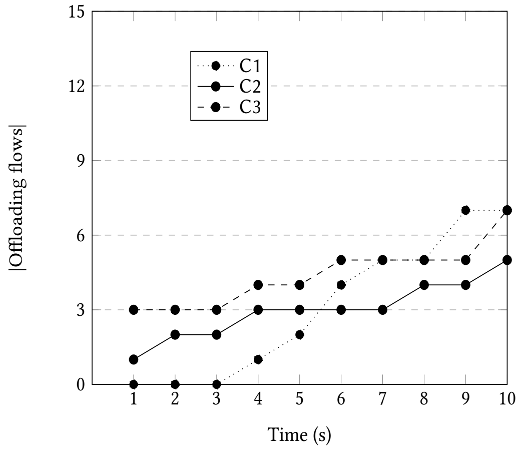

**Figure 4: Results of scenario #2 experiments**

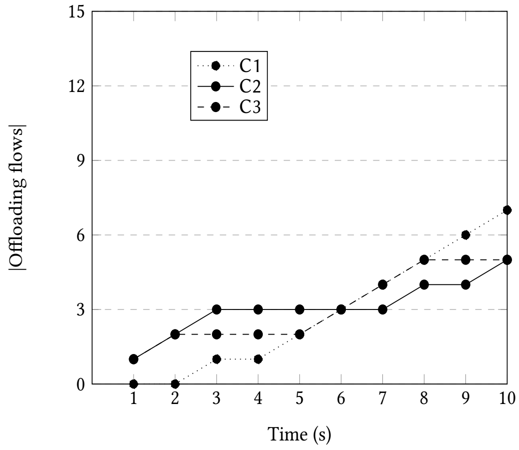

**Figure 5: Results of scenario #3 experiments**

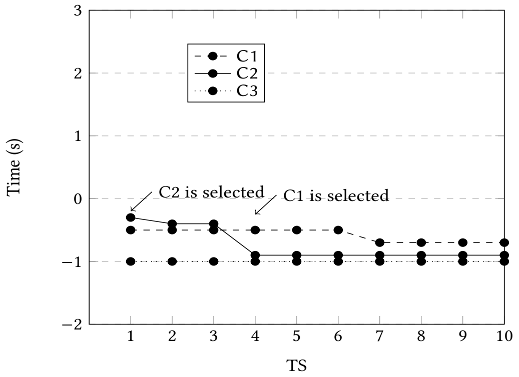

**Figure 6: Variation of $\mathcal{TS}$s measured by fog $F1$**

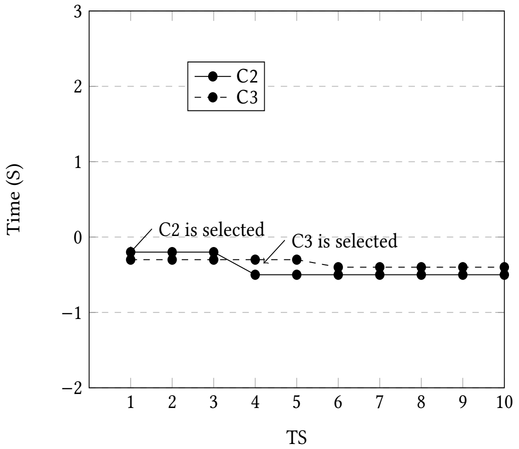

**Figure 7: Variation of $\mathcal{TS}$s measured by cloud $C1$**

data misuse cases, for example. To address these cases, a trust model has been designed and developed taking into account different factors namely, types of interactions between things, fogs, and clouds, recommendations of where things should send their data, legitimacy of data senders, quality-of-service of fogs/clouds, and quality-of-experience interacting with fog/clouds. The trust model has been demonstrated through a set of experiments.

In term of future work, we would like to complete the analysis of case 3 ($\mathcal{T}^i 2C^j_{\text{df}} \longrightarrow C^j 2C^k_{\text{of}}$) and case 4 ($\mathcal{F}^i 2C^j_{\text{df}} \longrightarrow C^j 2C^k_{\text{of}}$) as well as examine the impact of trust on developing a chain of offloading flows. By analogy with case 2, a chain illustrated with $1[\mathcal{F}^i 2\mathcal{F}^j_{\text{of}}]+ \longrightarrow \mathcal{F}^j 2C^k_{\text{of}} \longrightarrow 1[C^k 2C^l_{\text{of}}]+$ could be formed raising questions about the trustworthiness and traceability of who is offloading to whom.

## REFERENCES

[1] M.R. Abdmeziem, D. Tandjaoui, and I. Romdhani. In Anis Koubaa and Elhadi Shakshuki, editors, *Robots and Sensor Clouds*, chapter Architecting the Internet of Things: State of the Art. Springer International Publishing, 2016.

[2] M. Al-Khafajiy, T. Baker, H. Al-Libawy, Z. Maamar, M. Aloqaily, and Y. Jararweh. Improving Fog Computing Performance via Fog-2-Fog Collaboration. *Future Generation Computer Systems*, 100, 2019.

[3] P.M. Barnaghi and A.P. Sheth. On Searching the Internet of Things: Requirements and Challenges. *IEEE Intelligent Systems*, 31(6), 2016.

[4] F. Bonomi, R. Milito, P. Natarajan, and J. Zhu. Fog Computing: A Platform for Internet of Things and Analytics. In *Big Data and Internet of Things: A Roadmap for Smart Environments, Studies in Computational Intelligence.* Cisco, Springer International Publishing, 2014.

[5] M. De Donno, K. Tange, and N. Dragoni. Foundations and Evolution of Modern Computing Paradigms: Cloud, IoT, Edge, and Fog. *IEEE Access*, 7, 2019.

[6] S. Elnaffar, Z. Maamar, and Q.Z. Sheng. When Clouds Start Socializing: The Sky Model. *International Journal of E-Business Research*, 9(2), 2013.

[7] M. Fiedler, T. Hossfeld, and P. Tran-Gia. A Generic Quantitative Relationship between Quality of Experience and Quality of Service. *IEEE Network*, 24(2), 2010.

[8] J. Huang and D.N. Nicol. Trust Mechanisms for Cloud Computing. *Journal of Cloud Computing*, 2:9, 2013.

[9] K.M. Khan and Q. Malluhi. Establishing Trust in Cloud Computing. *IT Professional*, 12(5), 2010.

[10] K. Khebbeb, N. Hameurlain, and F. Belalab. A Maude-based Rewriting Approach to Model and Verify Cloud/Fog Self-Adaptation and Orchestration. *Journal of Systems Architecture*, 110, November 2020.

[11] X. Li. FASTCloud: A Framework of Assessment and Selection for Trustworthy Cloud Service based on QoS, 2020.

[12] Z. Maamar, T. Baker, N. Faci, E. Ugljanin, M. Al-Khafajiy, and V.A. Burégio. Towards a Seamless Coordination of Cloud and Fog: Illustration through the Internet-of-Things. In *Proceedings of the 34th ACM/SIGAPP Symposium on Applied Computing (SAC'2019)*, Limassol, Cyprus, 2019.

[13] Z. Maamar, T. Baker, M. Sellami, M. Asim, E. Ugljanin, and N. Faci. Cloud vs edge: Who serves the Internet-of-Things better? *Internet Technology Letters*, 1(5), 2018.

[14] R. Mahmud, K. Ramamohanarao, and R. Buyya. Application Management in Fog Computing Environments: A Taxonomy, Review and Future Directions. *ACM Computing Surveys*, 53(4), July 2020.

[15] J. Pakulski. Trust and Legitimacy. *Policy, Organisation and Society*, 5(1), 1992.

[16] M. Peter and G. Timothy. The NIST Definition of Cloud Computing. Technical Report 800-145, National Institute of Standards and Technology (NIST), September 2011.

[17] C. Puliafito, E. Mingozzi, F. Longo, A. Puliafito, and O. Rana. Fog Computing for the Internet of Things: A Survey. *ACM Transactions on Internet Technology*, 19(2), 2019.

[18] Y. Qin, Q.Z. Sheng, N.J.G. Falkner, S. Dustdar, H. Wang, and A.V. Vasilakos. When Things Matter: A Data-Centric View of the Internet of Things. *CoRR*, abs/1407.2704, 2014.

[19] M. Saidur Rahmana, I. Khalila, M. Atiquzzaman, and X. Yi. Towards Privacy Preserving AI-based Composition Framework in Edge Networks using Fully Homomorphic Encryption. *Engineering Applications of Artificial Intelligence*, 94, September 2020.

[20] G. Suchacka and J. Iwanski. Identifying Legitimate Web Users and Bots with Different Traffic Profiles - an Information Bottleneck Approach. *Knowledge Based Systems*, 197, 2020.

[21] M. Varela, L. Skorin-Kapov, and T. Ebrahimi. In S. Möller and A. Raake, editors, *T-Labs Series in Telecommunication Services*, chapter Quality of Service Versus Quality of Experience. Springer, Cham, 2014.

[22] B. Varghese, N. Wang, D.S. Nikolopoulos, and R. Buyya. Feasibility of Fog Computing. *arXiv preprint arXiv:1701.05451*, 2017.

[23] T. Wang, G. Zhang, M.Z. Alam Bhuiyan, A. Liu, W. Jia, and M. Xie. A Novel Trust Mechanism based on Fog Computing in Sensor-Cloud System. *Future Generation Computing Systems*, 109, 2020.

[24] F. Yahya, Z. Maamar, and K. Boukadi. A Multi-Criteria Decision Making Approach for Cloud-Fog Coordination. In *Proceedings of the 34th International Conference on Advanced Information Networking and Applications (AINA'2020)*, Caserta, Italy, 2020.

[25] J.P. Yoon and Z. Chen. Service Trustiness and Resource Legitimacy in Cloud Computing. In *Proceedings of the International Conference on P2P, Parallel, Grid, Cloud and Internet Computing (3PGCIC'2010)*, Fukuoka, Japan, 2010.

[26] Jing Yuan, Yu Zheng, Xing Xie, and Guangzhong Sun. Driving with knowledge from the physical world. In *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 316–324, 2011.