IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received June 26, 2018, accepted August 14, 2018, date of publication August 22, 2018, date of current version September 21, 2018.

Digital Object Identifier 10.1109/ACCESS.2018.2866491

Fog Computing: Survey of Trends, Architectures, Requirements, and Research Directions

RANESH KUMAR NAHA[1], SAURABH GARG[1], (Member, IEEE), DIMITRIOS GEORGAKOPOULOS[2], (Member, IEEE), PREM PRAKASH JAYARAMAN[2], (Member, IEEE), LONGXIANG GAO[3], (Senior Member, IEEE), YONG XIANG[3], (Senior Member, IEEE), AND RAJIV RANJAN[4], (Senior Member, IEEE)

[1]School of Technology, Environments and Design, University of Tasmania, Hobart, TAS 7001, Australia
[2]Faculty of Science, Engineering and Technology, Swinburne University of Technology, Melbourne, VIC 3122, Australia
[3]School of Information Technology, Deakin University, Burwood, VIC 3125, Australia
[4]School of Computing, Newcastle University, Newcastle upon Tyne NE1 7RU, U.K.

Corresponding author: Longxiang Gao (longxiang.gao@deakin.edu.au)

The work of R. K. Naha was supported by the University of Tasmania for providing Tasmania Graduate Research Scholarship.

ABSTRACT Emerging technologies such as the Internet of Things (IoT) require latency-aware computation for real-time application processing. In IoT environments, connected things generate a huge amount of data, which are generally referred to as big data. Data generated from IoT devices are generally processed in a cloud infrastructure because of the on-demand services and scalability features of the cloud computing paradigm. However, processing IoT application requests on the cloud exclusively is not an efficient solution for some IoT applications, especially time-sensitive ones. To address this issue, Fog computing, which resides in between cloud and IoT devices, was proposed. In general, in the Fog computing environment, IoT devices are connected to Fog devices. These Fog devices are located in close proximity to users and are responsible for intermediate computation and storage. One of the key challenges in running IoT applications in a Fog computing environment are resource allocation and task scheduling. Fog computing research is still in its infancy, and taxonomy-based investigation into the requirements of Fog infrastructure, platform, and applications mapped to current research is still required. This survey will help the industry and research community synthesize and identify the requirements for Fog computing. This paper starts with an overview of Fog computing in which the definition of Fog computing, research trends, and the technical differences between Fog and cloud are reviewed. Then, we investigate numerous proposed Fog computing architectures and describe the components of these architectures in detail. From this, the role of each component will be defined, which will help in the deployment of Fog computing. Next, a taxonomy of Fog computing is proposed by considering the requirements of the Fog computing paradigm. We also discuss existing research works and gaps in resource allocation and scheduling, fault tolerance, simulation tools, and Fog-based microservices. Finally, by addressing the limitations of current research works, we present some open issues, which will determine the future research direction for the Fog computing paradigm.

INDEX TERMS Fog computing, Internet of Things (IoT), fog devices, fault tolerance, IoT application, microservices.

I. INTRODUCTION

Individuals and organizations are increasingly becoming dependent on computers and smart devices to deal with daily tasks. These devices are generating data via various sensors and applications. As a result, organizations are generating and storing huge amounts of data on a regular basis [1]. After the proliferation of IoT, data generated by sensors has increased enormously. With this sudden increase in the volume of data being produced and inability of conventional databases to process various forms of structured and unstructured data, big data analytics has attained great attention in recent years. Every organization is now prioritizing the analysis of collected data to extract useful insights in order to make important decisions [2]. Nowadays, organizations need a dynamic IT infrastructure because of the shift to cloud computing due to its accessibility, scalability, and pay-per-use features. The most common services provided by the cloud are known as Software as a Service (SaaS), Platform as a Service (PaaS),

and Infrastructure as a Service (IaaS), all of which are heading towards Anything as a Service (XaaS) [3]. However, data generated from billions of sensors, referred to as big data, cannot be transferred and processed in the cloud. In addition, some IoT applications need to be processed faster than the cloud's current capability. This problem can be solved by using the Fog computing paradigm, which harnesses the processing power of devices located near users (idle computing power) to support utilization of storage, processing, and networking at the edge [4].

Fog computing is a decentralized computing concept, which does not exclusively rely on any central component like cloud computing [5], [6]. It is able to overcome the high latency problem of the cloud by using idle resources of various devices near users. However, Fog computing relies on the cloud to do complex processing. Unlike cloud computing, Fog computing is a decentralized computing concept, where the many devices around us, which have computation capacity, are utilized. Currently, even a low- specification smartphone has processing capacity, sometimes with multiple cores. Hence, many devices like smartphones, switches, routers, base stations, and other network management devices equipped with processing power and storage capacity can act as Fog devices. The resources of these devices are idle outside of peak hours.

Many research issues relating to Fog computing are emerging due to its ubiquitous connectivity and heterogeneous organization. In the Fog computing paradigm, key issues are the requirements and the deployment of Fog computing environment. This is because the devices that exist in Fog environments are heterogeneous: therefore, the question that arises is how will Fog computing tackle the new challenges of resource management and failure handling in such a heterogeneous environment? Hence, it is necessary to investigate the very basic requirements for all other related aspects such as deployment issues, simulations, resource management, fault tolerance, and services. Several reviews [5], [7]–[12] have been done on Fog computing. Here, we present the focus and survey domains of these review works in brief.

Similar concepts of Fog computing, definitions, application scenarios, and numerous issues are described by one study [7]. Hu *et al.* [8] presented the hierarchical architecture of Fog computing and technologies like computing, communication, and storage technologies, namely resource management, security, and privacy protection that support Fog deployment and application. Baccarelli *et al.* [9] surveyed Fog computing and the Internet of Everything (IoE) with an integrated point of view of Fog computing and IoE. Varshney and Simmhan [10] reviewed various dimensions of application characteristics, system architecture, and platform abstractions of edge, Fog, and cloud ecosystems. Perera *et al.* [11] reviewed the Fog computing domain from the platform perspectives of developers and end users towards building a sustainable sensing infrastructure for smart city applications. Mahmud *et al.* [5] presented a taxonomy of Fog computing according to the identified challenges and its key features. The proposed taxonomy provides a classification of the existing works in Fog computing. Mouradian *et al.* [12] reviewed Fog architecture and algorithms based on six different evaluation criteria, namely heterogeneity, QoS management, scalability, mobility, federation, and interoperability. However, none of the studies had investigated taxonomy based on the requirements of infrastructure, platform, and application in Fog computing. Moreover, none of them comprehensively investigated fault tolerance, resource management, or microservices in Fog computing. We consider the aforementioned current issues and discuss these extensively and also highlight how cloud computing-related solutions could be employed in the Fog in some cases. The contributions of this review work can be summarized as follows:

- Present the research trends in Fog computing by investigating the number of published research works and search occurrences in Google Scholar.
- Review of several Fog computing architectures and presentation of a detailed architecture, as most of the previous researchers only presented high-level architecture.
- Present a taxonomy by considering the requirements of infrastructure, platform, and application in the Fog computing paradigm.
- Identify Fog computing research gaps in resource allocation and scheduling, fault tolerance, simulation tools, and Fog-based microservices.
- Address the limitations of current research works and some open issues in infrastructure, platform, and applications.

From this survey, the industry and research community will be able to gain insight into the requirements for building a Fog computing environment with a better understanding of resource management in the Fog.

The remainder of this paper is organized as follows: Section II surveys definitions and research trends in Fog computing. A technical comparison between Fog and cloud paradigms presents in section III. Section IV discusses computing paradigms similar to Fog computing. Section V presents related works on Fog computing architecture and discusses the components of the Fog computing architecture. Section VI shows the taxonomy of Fog computing by reviewing its requirements. Section VII presents various application dimension of Fog computing. Section VIII discusses current state-of-the-art Fog computing technology. Section IX presents open issues and future research direction. Section X concludes the paper.

II. OVERVIEW OF FOG COMPUTING

The term 'Fog computing' was proposed in 2012 by researchers from Cisco Systems [13]. Processing application logic and data at the edge is not a new concept. The concept of Edge computation emerged around the 2000s [14], [15] and another similar concept, cloudlets, was introduced in 2009 [16]. Both Cloudlets and Fog computing

are the advancements of a similar concept, which revolves around processing at the edge level. While cloudlets are applied in the mobile network, Fog computing is applied to connected things such as IoT, which plays into the concept of IoT [17].

Fog is both a virtualized and non-virtualized computing paradigm that provides networking, storage, and computation services amid cloud servers and IoT devices [4], [13]. However, these services are not completely located at the network edge. The Fog is a distributed computing approach that mainly focuses on facilitating applications, which require low latency services [18], Fog computing also supports non-latency aware services. It is obvious that using idle computation resources near the users will improve overall service performance, if the volume of processing were not that high. A huge number of heterogeneous nodes will be connected to the Fog. These nodes include sensors and actuators among others [13]. Computation is performed in Fog devices when necessary and storage facilities are also available for a short period of time, at least in most Fog devices. Time-sensitive computation in the Fog is done without the involvement of third parties, and in most cases, is done by the Fog processing devices. According to Yi *et al.* [7], the Fog computing paradigm supports the running of new services or basic network functions and applications in a sandboxed environment similar to cloudlets. However, the subject is still a research challenge because the question of how the Fog will provide these service still remains. In addition, will the Fog have cloud service providers or will it be like a single entity as a whole? Figure 1 shows a basic model of Fog Computing. Fog devices, Fog servers, and gateways are the basic computation components in the Fog environment. Any device that has computation, networking, and storage capabilities can act as a Fog device. These devices include set-top boxes, switches, routers, base stations, proxy servers or any other computing device. Fog servers that manage several Fog devices and Fog gateways are responsible for translation services between heterogeneous devices in the Fog computing environment. Fog gateways also provide translation services between IoT, Fog, and cloud layers. New challenges in this emerging computing paradigm have emerged in the past couple of years.

In this section, we discuss the various definitions of Fog computing and define Fog computing from our point of view. In addition, we discuss and analyze research trends in Fog computing. Finally, we compare the technical differences between Fog computing and cloud computing.

A. DEFINITION OF FOG COMPUTING

Fog computing is a distributed computing paradigm where processing is done at the edge of the network with seamless integration of the cloud infrastructure. It enables a computing facility for IoT environments or other latency sensitive application environments. It is estimated that about 50 billion "things" will be connected to the Internet by 2020 [19]. Transferring all data from all connected devices for processing on the cloud will need massive amounts of bandwidth

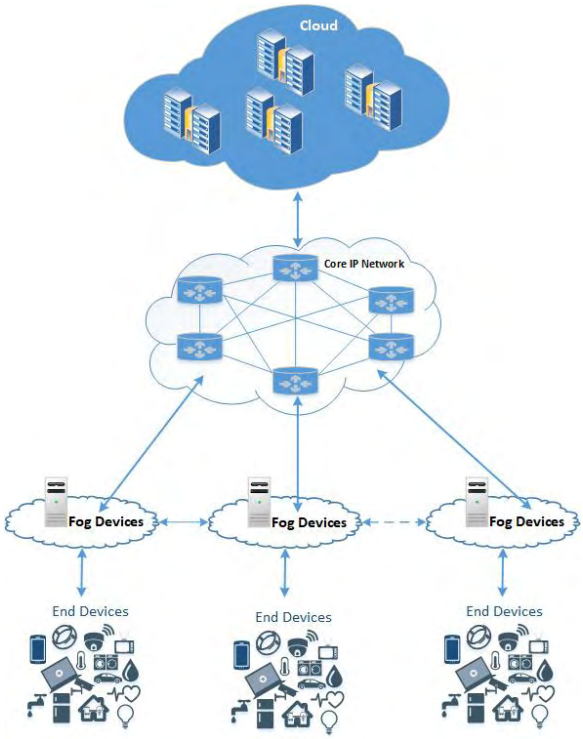

FIGURE 1. A model of Fog computing.

and storage. All devices are not connected to the controller via IP but connected by some other IoT industrial protocols. Because of this, a translation process is also needed for the processing or storing of information from IoT devices. Various researchers have defined Fog computing in different ways. Some examples are as follows:

- *"Fog computing is a highly virtualized platform that provides compute, storage, and networking services between IoT devices and traditional cloud computing data centers, typically, but not exclusively located at the edge of network." [13]*
- *"Fog computing is a scenario where a huge number of heterogeneous (wireless and sometimes autonomous) ubiquitous and decentralised devices communicate and potentially cooperate among them and with the network to perform storage and processing tasks without the intervention of third parties. These tasks can be for supporting basic network functions or new services and applications that run in a sandboxed environment. Users leasing part of their devices to host these services get incentives for doing so." [20]*
- *"The term Fog computing or Edge Computing means that rather than hosting and working from a centralized cloud, Fog systems operate on network ends. It is a term for placing some processes and resources at the edge of the cloud, instead of establishing channels for cloud storage and utilization." [21]*

The first definition of Fog computing was presented by Bonomi *et al.* [13], where they addressed the computing

paradigm as a highly virtualized platform. However, some IoT devices such as smartphones are not virtualized but could also be a part of the Fog infrastructure, as some processing could still be done. According to Cisco [22], the Fog computing paradigm provides an ideal place to analyze most data near the devices that produce and act on that data instantaneously. The Fog is located near things that are able to process and act on the data generated. The devices that are within the Fog environment are known as Fog devices. These nodes can be deployed at any place with a connectivity to the network: on the power pole, on the factory floor, alongside the road, alongside the railway line, in a vehicle, inside a shopping mall, on an oil rig, etc. A device that has processing, storage, memory, and network capability can act as a Fog device. Although the Fog extends the cloud, technically it resides in between the cloud and IoT devices and handles processing and storage tasks in close proximity to the user. Yi *et al.* [7] stated that the definition given by Vaquero and Rodero-Merino [20] is debatable and a definition that can distinguish clearly between Fog computing and other related computing paradigms is still required. The definition given by IBM [21] represents Edge and Fog computing as the same computing paradigm. According to Shi *et al.* [23], Fog computing focuses more on the infrastructure side while edge computing focuses more on the things' side. Furthermore, Edge computing is not spontaneously associated with any cloud-based services such as SaaS, IaaS, and PaaS [5]. In brief, Table 1 summarizes Fog definitions provided by various research works.

Considering the above definitions, we define Fog computing as follows:

- *Fog computing is a distributed computing platform where most of the processing will be done by virtualized and non-virtualized end or edge devices. It is also associated with the cloud for non-latency-aware processing and long-term storage of useful data by residing in between users and the cloud.*

In our definition, we considered all devices with computing and storage capacity as Fog devices and also more precisely identified the role of the cloud in the Fog computing environment.

TABLE 1. Summary of Fog computing definitions.

Defined by	Characteristics
Bonomi et al. [13]	Highly virtualized
	Reside between IoT devices and cloud
	Not exclusively located at the edge
Cisco Systems [22]	Extends the Cloud
	Generally used for IoT
	Can be deployed anywhere
	Fog device consists of processing, storage, and network connectivity
Vaquero and Rodero-Merino [20]	heterogeneous, ubiquitous and decentralised devices communication
	Storage and processing done without third party invention
	Run in a sandboxed environment
	Leasing part of users devices and provide incentive
IBM [21]	Defined Fog and Edge computing as similar
	Not depends on centralized cloud
	Resides at network ends
	Place some resource and at the edge of the cloud
Proposed Definition	Virtualization and non-virtualization characteristics
	Association with the cloud for non-latency-aware processing and storage
	Any edge device with available processing power and storage capability can be act as a Fog device
	Always resides between end users and cloud

B. FOG COMPUTING RESEARCH TRENDS

Growing attention towards processing data closer to the users has been observed among industries and the academia in the past few years. Handling IoT-generated data at the edge level will help improve overall processing time. In this section, we investigate Fog and other related technological trends for the past few years in the research community. According to the Gartner hype cycle, in July 2017 [24], the peak emerging technology is the smart home, which would perform better with the incorporation of the Fog computing environment. A Hype Cycle [24] represents common patterns of new trending technologies. Fog computing can also enable latency-aware smart home services in a more efficient and convenient way, especially for emergency response smart home applications. According to the Gartner hype cycle demonstration, some other influencing technologies include virtual assistants, autonomous vehicles, IoT platforms, smart robots, edge computing, and smart workspaces, which are required to support latency-aware applications. All these mentioned technologies could benefit from the support of the Fog computing paradigm due to latency sensitiveness, connectivity to the cloud, and edge-level data processing capability. Except for the autonomous vehicle technology, all other aforementioned technologies will reach the market adoption threshold in the next 10 years. Besides the hype cycle analysis, we analyzed the search occurrence of Fog and other related technologies in Google Scholar. In addition,

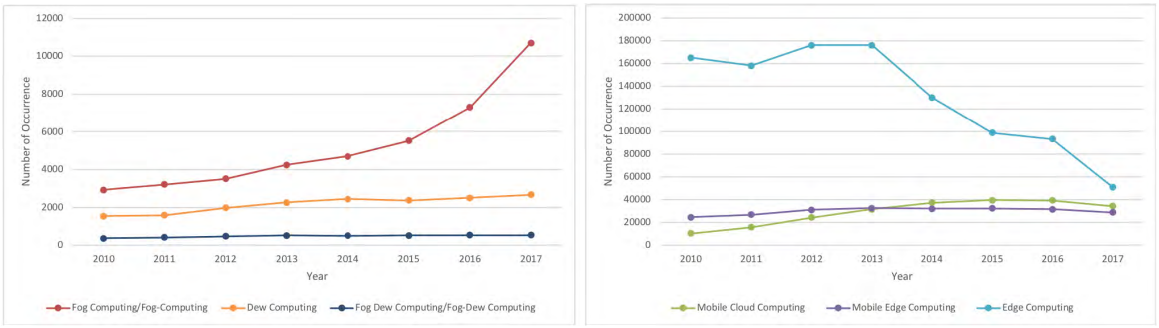

FIGURE 2. Search occurrence of similar technologies like the Fog in Google Scholar.

the number of papers available in different digital libraries related to the Fog was also analyzed.

Google Scholar search occurrences of various similar technologies to Fog were investigated in the past few years, as presented in Figure 2. According to the data, edge computing is the topmost searched item in Google Scholar compared to other similar technologies. However, the search trend decreased by more than three times in the past eight years for edge computing. Mobile cloud computing and mobile edge computing are the other two top-searched computing paradigm after edge computing. The lowest trend observed was for dew computing and Fog dew computing. While the trend for edge computing is decreasing, Fog computing related to scholarly searches is increasing year by year, and has increased by 2.5 times from 2010 to 2017. This shows that Fog computing is the fastest growing research area in academia and will have a great impact on the industry as well.

Fog computing topic search in the Web of Science shows that the number of scholarly articles has more than doubled between 2015 and 2016, as per Figure 3. The first paper with 'Fog computing' in its title was published in 2012. Since then, about 564 journal and conference articles have been published on this topic in the four major digital libraries (Web of Science, Science Direct, IEEE Xplore, and ACM), as presented in Figure 4. Cloud computing first emerged in 2008 [25]. This shows that Fog computing publications have dramatically increased, as no study in this area was seen in the couple of years following the introduction of cloud computing research in 2008 (see Figure 5).

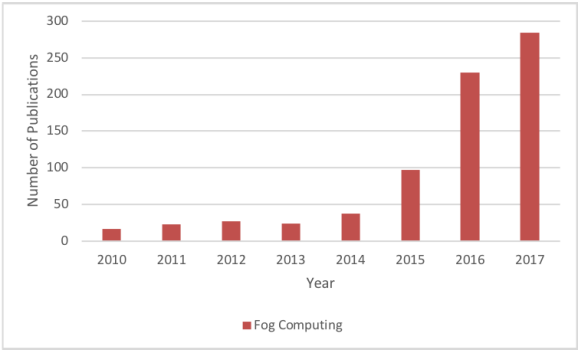

FIGURE 3. No. of Fog computing-related papers in the Web of Science (as Feb 2018).

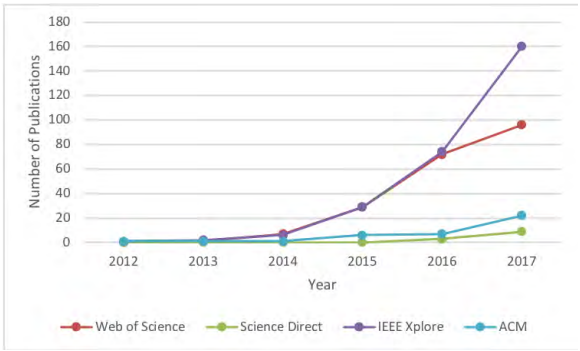

FIGURE 4. Number of publications with "Fog computing" in the title in the four major digital libraries.

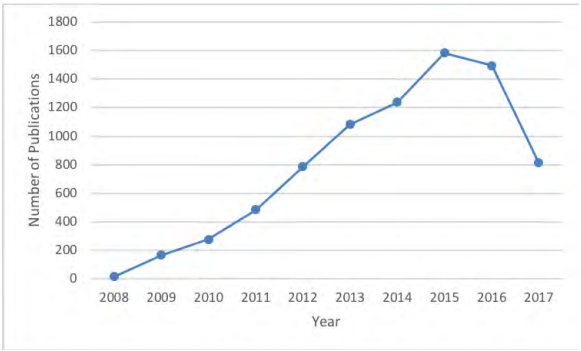

FIGURE 5. Published articles with the title cloud computing in the Web of Science.

From our observation, it is obvious that the interest in Fog computing research is rapidly increasing. Idle resources in the form of devices near users can be utilized within the Fog computing concept. Thus, a clear direction to market the adoption and technological development of Fog deployment has emerged.

III. DIFFERENCE BETWEEN FOG AND CLOUD COMPUTING PARADIGM

Fog computing architectures are based on Fog clusters where multiple Fog devices participate to cooperate with the processing. On the other hand, datacenters are the main physical components of clouds. Because of this, cloud computing has

TABLE 2. Technical difference between Fog and cloud.

	Fog	Cloud
Participating Nodes	Constantly dynamic	Variable
Management	Distributed/Centralized	Centralized
Computation device	Any device with computation power	Powerful Server System
Nature of Failure	Highly diverse	Predictable
Connectivity from user	Mostly wireless	High speed (With the combination of wire and wireless)
Internal connectivity	Mostly wireless	Mostly Wired
Power source	Battery/Direct power/Green Energy, such as solar power	Direct power
Power consumption	Low	High
Computation capacity	Low	High
Storage capacity	Low	High
Network latency	Low	High
Node mobility	High	Very low
Number of intermediate hop	One/Few	Multi
Application type	latency-aware	Non latency-aware
Real time application handling	Achievable	Difficult
Computation cost	Low	High
Cooling cost	Very low	High
Space required for deployment	Very little, also possible to install at outdoor on existing infrastructure	Warehouse size building

high operational costs and energy consumption. By comparison, energy consumption and operation costs in the Fog computing paradigm is low. The Fog is located closer to the user, so the distance between users and Fog devices could be one or a few hops, which is also agreed by Hu *et al.* [8]. However, according to Mahmud *et al.* [26], the distance between users to the Fog is one or two hops. Again, Luan *et al.* [27] argued that the distance should be one hop with wireless connectivity. Yet, all agreed with the distance between the users to the cloud, which is a multi-hop distance. Due to the distance, communication latency for the cloud is always high compared to the Fog. The cloud is a more centralized approach while the Fog is a more distributed approach based on geographical orchestration [26].

Real-time Interaction is not possible for the cloud due to its high latency, but this problem can be easily resolved by Fog computing. On the other hand, the rate of failure in the Fog is high because of wireless connectivity, decentralized management, and power failure [26], [28]–[30]. Most devices in Fog environments will be connected wirelessly since smart gadgets and handheld devices will be participating in Fog systems [31]. These devices, and other network management devices, are mostly decentralized. These devices could fail when software is not managed correctly. Users may not be aware of malicious software that could lead to device failure. Moreover, Fog processing could fail in other cases as well, for example, each Fog device is responsible for performing its own application processing. So, the IoT application processing in a Fog device always takes second priority. If the Fog device is fully utilized by the application of the device itself, then it will fail to do any Fog processing. Hence, the scheduling of applications and resources in the Fog is more complex. In addition, failure handling in the Fog is competitive because of power failure, which is only an issue because the devices run on battery power. Altogether, Table 2 shows the technical differences between the cloud and the Fog.

Definitely, it cannot be said that the Fog can replace the cloud. We cannot even conclude that the Fog is better than the cloud either, both contribute differently via fulfilling different perspectives and requirements.

IV. RELATED PARADIGMS AND TECHNOLOGIES

Fog computing uses computing resources near underlying networks, located between the traditional cloud and edge devices, to provide better and faster application processing and services [13]. Several similar computing paradigms exist besides Fog computing such as Mobile

Cloud Computing (MCC), Mobile-Edge Computing (MEC), Edge Computing, Dew Computing, and Fog-dew computing. In cloud computing, all IoT devices are directly connected to the cloud and computation totally depends on the cloud. However, all the above similar technologies do not exclusively depend on the cloud, but depend on some intermediate devices for computation; some of them do not even require a connection to the cloud. Figure 6 shows the high-level architecture of these technologies.

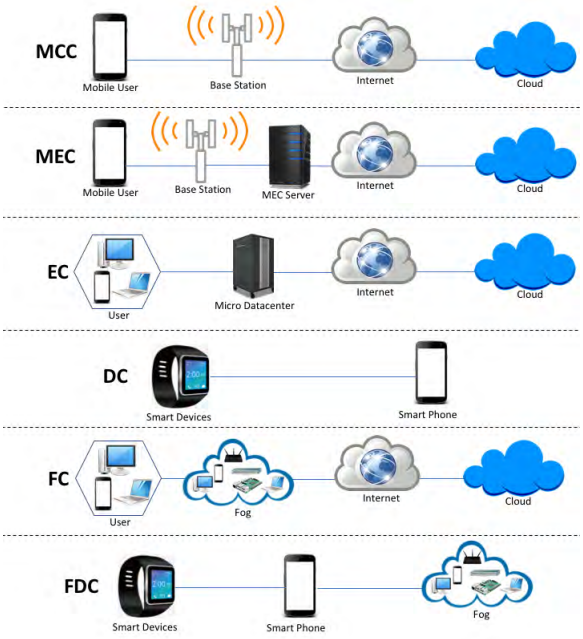

FIGURE 6. High level architecture of Mobile Cloud Computing (MCC), Mobile Edge Computing (MEC), Edge Computing (EC), Dew Computing (DC), Fog Computing (FC) and Fog Dew Computing (FDC).

A. MOBILE CLOUD COMPUTING (MCC)

Remote execution of offloaded mobile services is done with the support of MCC near end users [32], [33]. MCC overcomes the computational, energy, and storage resource limitation of smart mobile devices. Generally, a lightweight cloud server (cloudlet) is placed at the edge of the network [34] to overcome these issues. MCC is a mobile computing technology, which provides unrestricted functionality, mobility, and storage facility through heterogeneous network connectivity. This technology also provides unified elastic computing resources by following the pay-per-use model. It also provides access to data, application, and cloud via the Internet for mobile users. It is expected that this technology will be applied in education, urban and rural development, healthcare, and more realistic social networking in the future [32]. Nowadays, many computation-intensive applications are widely available, such as Augmented Reality, computer vision and graphics, speech recognition, machine learning, planning and decision-making, and natural language processing applications. However, simply designing powerful mobile devices will not meet the requirements for these applications [33]. Rather, the applications require edge processing as well as collaboration with the cloud for complex processing. Thus, mobile computing demands fundamental changes to cloud computing, for example, a low-latency middle tier, programming models to enable seamless remote execution, basic mobile cloud services such as presence services, cloud infrastructure optimization for mobile applications, memcache services, and so on [33]. The convergence of mobile cloud computing is predicated on a reliable, end-to-end network, and high bandwidth, which isdifficult to guarantee in harsh environments. One of the solutions to this deep-rooted problem is the VM-based cloudlets located at a closer location to the mobile device [34].

B. MOBILE EDGE COMPUTING (MEC)

MEC proposes the co-location of computing and storage resources at the base stations of cellular networks [35]. MEC could either be connected or disconnected to cloud datacenters in a remote location. Hence, MEC supports two- or three-tier hierarchical application deployments along with end mobile devices [36]. In a MEC ecosystem, a new device called the MEC server needs to be deployed near base station towers to provide processing and storage capabilities at the edge. Four participants are involved in this computing paradigm, which are the mobile end users, network operators, Internet infrastructure provider (InPs),and application service provider. Mobile end users are the main consumer of the system and request their service via user equipment (UE). Network operators manage and maintain the operation of base stations, mobile core network, and MEC servers. InPs maintain Internet connectivity and routers. Application service providers host the application services in the content delivery networks (CDN) or within a data centers. Processing of requests from the UE will search out the closest MEC. The MEC server is capable of processing user request instead of forwarding it to remote Internet services. In a case where it is not possible to process or complete a request at the MEC sever; the request will be forwarded to remote CDNs or data centers [35].

According to Klas [36], MEC is the evolution of mobile base stations. It is a natural development. It is a collaborative deployment of telecommunication and IT networking. This computing paradigm enables new vertical business segments and services to individual end users and enterprise consumers. Various services could be delivered through this computing paradigm including IoT, location services, augmented reality, caching service, video analytics, and local content distribution. It can deliver real-time low-latency access of local content or by caching content at the MEC server. However, the main limitation of this system is the installation of the MEC server, which is specifically dedicated to MEC services. Scaling is another big issue with the increase in resource demand over time.

C. EDGE COMPUTING

Edge devices or edge servers provide computation facilities in Edge computing. In general, edge computing does not spontaneously associate with any types of cloud-based

services and concentrates more on the IoT device side [23]. One study defined the edge as any network or computing resource near the path between cloud data centers and data sources [23]. Any smart device or sensor could have data sources but the edge is different. For example, a cloudlet and a micro datacenter is the edge of the mobile application and cloud, whereas the IoT gateway is the edge between IoT sensors and cloud. Similarly, if a cloud application is running on a smartphone, then the smartphone is the edge of the application and the cloud [37]. The main motivation of edge computing is that the computation should be done at a closer location to the data sources.

In the edge computing concept, things are not only consuming data but also produce data by taking part in processing. Edge devices can perform computation task from the cloud besides requesting services and content. Data storage, computing offloading, processing, and caching will be done by an edge node. The edge device is also capable of distributing requests and providing service on behalf of the cloud to the users. In such scenarios, edge devices need to be well designed to meet privacy requirements, reliability measures, and security concerns [23].

D. DEW COMPUTING (DC)

In the current computing hierarchy, Dew Computing [38] is situated at the ground level of the cloud and Fog computing environment [39]. DC goes beyond the concept of service, storage, and network, to a sub-platform, which is based on a microservice concept for which its computing hierarchy is vertically distributed [39]. The DC approach facilitates resources such as sensors, tablets, and smartphones that are seamlessly connected to a network. Because of this, DC covers a wide range of ad-hoc-based networking technologies [39].

Skala *et al.* [39] argued that DC is much more useful in everyday life compared to Fog computing. Fog supports IoT-based applications, which demand less latency and real-time capability and a dynamic network configuration while DC is microservice concept and thus is not dependent on any centralized device, server, or cloud. They provide an example in which DC could be integrated into a smart traffic control system, where data collection and processing units will be located in between the traffic signals. These units generate a collective overview of the current traffic conditions. In such a way, a car with low fuel will be notified before entering heavy congestion, or a hybrid car will be informed of switching to conventional fuel before approaching the congestion. As a result, cars with less fuel will be rescued from unwanted situations and hybrid cars could reduce exhaust smoke densities significantly. Although the concept is microservice-based, the processing is completed in Fog devices. In the Fog computing concept, it is not crucial that applications must be dependent on the cloud or require the storing of results in the cloud. On the other hand, if such traffic processing information were stored, it would help strategic decision-making to improve traffic management. Dew computing is an emerging research area and its goal is to use the full potential of cloud and local resources [40].

E. FOG-DEW COMPUTING

In the architecture of Fog-dew computing, IoT devices need not have an active Internet connection while being connected to the community server. The community server will interact with the cloud and is responsible for providing services to the IoT devices [41].

Cloud computing always needs an Internet connection, which is the main drawback of the cloud. While the cloud is unable to serve users without an Internet connection, Fog-dew computing facilitates offline services without an Internet connection. However, there are some exceptions. For example, the navigation app, Waze, allows users to navigate offline. This feature was also recently added to Google Maps. In this case, a map information file for a specific area is downloaded to the user device and allows users to navigate during an offline state. Another example is Google Drive and Dropbox, where users can delete, create, and update files and folders in offline mode and then sync once the device is connected to the Internet. However, these services are not purely offline-we may not rely on the Internet directly but we cannot completely ignore Internet connection. The situation becomes more complex when a single user uses multiple offline devices alongside the complexity that arises in a multiuser environment. Such situations could be mitigated with the help of Fog-dew computing.

In summary, in the Fog computing paradigm, IoT devices are connected to the cloud via Fog devices. Fog devices are connected to the cloud through the core network. Fog computing is a combination of MEC and MCC [7] but the main goal of all Fog-related paradigms is to perform processing at the edge. These related paradigms differ from each other based on Internet and cloud connectivity. Also, the amount of processing that needs to be done at the edge differs based on service requirements. Furthermore, the type of devices that will be used for computation and storage purposes is also another issue. In summary, Table 3 shows the characteristics of the above-discussed related computing paradigms along with the Fog computing paradigm.

V. ARCHITECTURE OF FOG COMPUTING

For market adoption and deployment, Fog computing must have a standard architecture. There is no available standard architecture to date. However, many research works have presented Fog computing architectures. In this section, we firstly discuss the high-level architecture of Fog computing. Furthermore, we summarize some proposed architectures for Fog computing. Finally, we present a detailed architecture for Fog computing with a comprehensive description of each component of the architecture.

A. HIGH-LEVEL ARCHITECTURE OF FOG COMPUTING

In high-level architecture, the Fog computing paradigm has three different layers, as shown in Figure 7. The most

TABLE 3. Summary of similar technologies like Fog.

Computing Paradigm & Applications	MCC	MEC	EC	DC	Fog-DC	Fog
	Offload mobile applications to the computation unit closer to the user					
	Mobile apps	Mobile apps	IoT related apps	Smart Devices (Fitbit, health monitoring)	Smart Devices (Fitbit, health monitoring)	IoT apps, Mobile apps, Video streaming, smart grid, smart transportation system, big data processing, stream processing
Connection to the Cloud	Yes	Yes or No	Yes or No	No	No	Yes
Types of Users	Mobile	Mobile	Mobile / Stationary	Smart sensor based devices	Smart sensor based devices	Mobile / Stationary
Virtualization Technology	Hypervisor / Container	Hypervisor / Container	Hypervisor / Container	Container	Container	Hypervisor / Container
Main Computation Element	Base station server	MEC server	Micro Data Center	Smartphone	Smartphone	Any device with the capabilities of computation, storage, memory and network adapter
Example of Commercial Prototype	Akamai [42]	-	Ec-IoT (Huawei) [43]	-	-	IOx (Cisco) [44]
Example of R&D Prototype	-	Hyrax [45], Saguna Open-RAN [46]	-	-	-	-

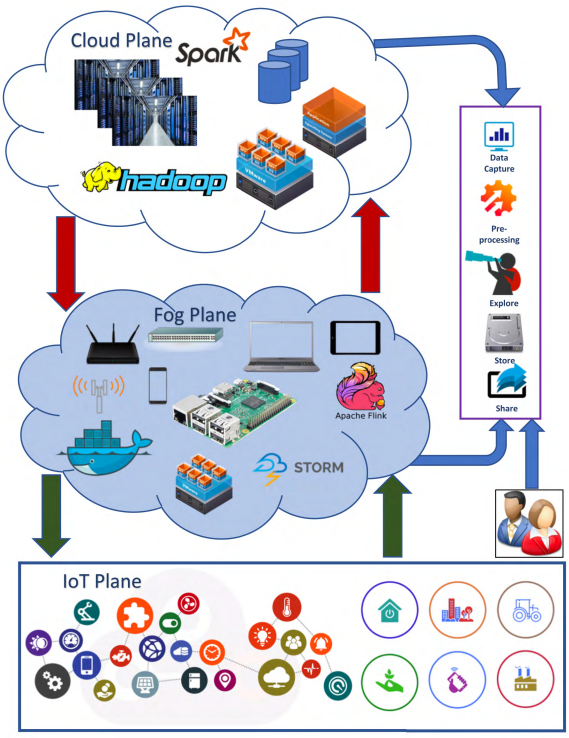

FIGURE 7. High level Fog computing architecture.

important layer is the Fog layer. This layer consists of all intermediate computing devices. Traditional virtualization technologies can be used at this plane, similar to the cloud. However, considering the resource availability, employing container-based virtualization is more appropriate. This layer accumulates sensor-generated data from the IoT layer and sends an actuation-related request after processing. Although it seems that the big data problem is solved by processing generated data at the edge level, billions of devices will create big data issue. In fact, it is possible to employ small- and medium-scale big data processing at this level. Many research works have been undertaken to process big data in the Fog plane [47]–[53].

The bottommost layer is the IoT plane, which consists of all connected devices. The devices on this plane perform the sensing and actuation process. For time-sensitive applications, processing should be done on the Fog plane exclusively while the cloud can perform other processing that is not time-sensitive. However, the Fog layer will manage what needs to be sent to the cloud and what should not. The users are able to get services from both the Fog and cloud based on their request. However, the cloud plane will manage complex processing and storage.

B. VARIOUS PROPOSED ARCHITECTURES FOR FOG COMPUTING

Layered representation is the best way to represent Fog architecture. Many works have been done to quantify the layer-based concept of Fog architecture [4], [27], [54]–[58]. From our review, we found that researchers have proposed three [27], [56]–[58], four [55], five [4], and six [54] layers in the Fog architecture.

Everyone has their own justifications for their claims. If we ignored the user plane, it is obvious that Fog architecture could be defined as three different levels from the high level. As we proceed to the more implementation-type level, the number of layers in the architecture would vary, giving rise to five [4] and six [54] levels in the Fog computing layer.

Aazam and Huh [54] presented six different layers based on specific tasks. On the other hand, Dastjerdi *et al.* [4]

defined five different layers based on network perspective. Other high-level architectures in Fog computing were also presented by various researchers including the hierarchical Fog architecture [59], [60], OpenFog architecture [61], Fog network architecture [62], Fog architecture for Internet of energy [9], Fog computing Architecture based on nervous system [63], and IFCIoT architecture [64]. After reviewing the literature stated above, we define the components of Fog computing architecture, which is presented in the following subsection.

C. COMPONENTS OF FOG COMPUTING ARCHITECTURE

Fog computing architecture consists of several layers. Each layer and its components are shown in Figure 8. In this subsection, we discuss various components of the Fog computing architecture. The components are divided into several groups based on their functionality, which is defined as the layer. These functionalities will enable IoT devices to communicate with various Fog devices, servers, gateways, and the cloud. A detailed explanation of each layer is given below, where a smart transportation use case is considered in the explanation.

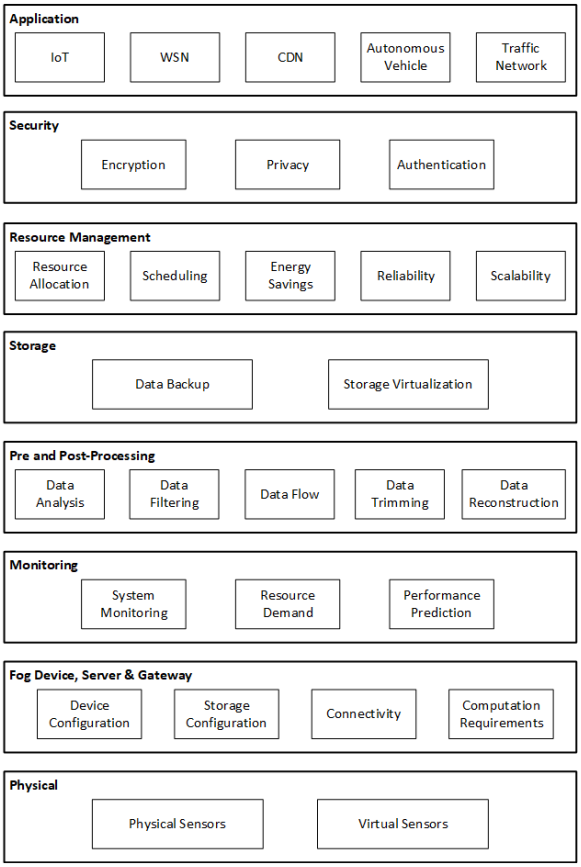

FIGURE 8. Components of Fog computing architecture.

1) PHYSICAL LAYER

The basic data source for Fog computing is the various forms of data emitted by the sensors [57]. These data could be generated from smart devices, temperature sensors, humidity sensors, smart homes, the CCTV surveillance system, traffic monitoring system, self-driving vehicles, and so on. For instance, if we wanted to implement a smart traffic management and monitoring system, we need to get updated traffic conditions of all roads from various sensors, roadside devices, and cameras, which will help manage traffic signals. It is also necessary to predict future traffic demand by collecting data from various GPS sensors. Besides physical sensors, the role of virtual sensors is also important [54], if a road accident occurred, it would not be possible to decide using just a single sensor whether the road should be blocked or traffic should keep going. The road might have one or more lanes-one lane may be affected by this occurrence while another lane could enable the traffic flow to continue, but the traffic handling capacity will be decreased due to this occurrence. In this case, a virtual sensor might help obtain an immediate decision on road conditions, traffic multiplexing, and traffic rerouting. Hence, the physical layer consists of physical and virtual sensors, where any data generation device could fall into any of these groups.

2) FOG DEVICE, SERVER, AND GATEWAY LAYER

The Fog device, Fog server, or Fog gateway could be a standalone device or an IoT device [57], [59], [62]. However, it is obvious that the Fog server should have a higher configuration than the Fog device and gateway since it manages several Fog devices. Various factors are involved so that the Fog server can run. These include its role, hardware configuration, connectivity, number of devices it can manage, and so on. Whether the Fog server is distinct or part of an IoT device is defined by its role. A group of physical and virtual sensors will be connected to a Fog device. Similarly, a group of Fog devices will be connected to a Fog server. In this context, the Fog server should have higher processing and storage capacity compared to the Fog device. A specific cluster of Fog devices, which are connected to the same server, can communicate with each other when required. In the smart transportation use case, some application processing might depend on other Fog clusters. For example, if an application needed to find a fuel-efficient route, it might need information about other sensor clusters or Fog device clusters. To reach an appropriate decision, processing needs to be done in multiple Fog devices and servers. The Fog server and device layer are responsible for managing and maintaining information on hardware configuration, storage configuration, and connectivity of device and servers. This layer also manages the computation requirements requested by various applications. Computation requirements depend on data flow and the total number of IoT devices connected to the Fog device, as well as the total number of Fog devices connected to the Fog server. The communication between several Fog servers is maintained by this layer. For example, a Cisco IOx-supported 800 series router can be used as a Fog device and Cisco Fog data service devices can be used as the Fog server [65], [66].

3) MONITORING LAYER

The monitoring layer always keeps track of the system performance and resources [54], services, and responses. System monitoring components help choose the appropriate resources during operation. Various applications run in smart transportation system scenarios. Therefore, it is obvious that a situation could arise when resource availability will be negative for computation or storage on a Fog device. A similar case could happen to the Fog server. To tackle these kinds of situations, the Fog device and servers will seek help from other peers. Thus, the system monitoring component will help decide such things efficiently. The resource demand component monitors current resource demand and can predict future demand for resources based on current resource usage and user activities. In this way, the system will be able to deal with any awkward situations where resource outage might occur. Performance prediction monitors can predict Fog computing performance based on system load and resource availability. This component is required to maintain appropriate QoS requirements in service level agreements. If SLA violation occurs frequently, then the cost of the system for the provider will be increased because of the penalty. Although performance prediction cannot eliminate this issue completely, it will be able to minimize overall SLA violation by predicting the performance and usage of the system.

4) PRE AND POST-PROCESSING LAYER

This layer contains multiple components, which specifically work on basic and advanced data analysis. At this level, acquired data are analyzed and filtered, and data trimming and reconstruction are also done when necessary. After processing the data, the data flow component decides whether the data needs to be stored locally or should be sent to the cloud for long-term storage [59]. The main challenge in Fog computing is to process data at the edge and minimize the volume of data that needs to be stored; this phenomenon is referred to as stream processing. In the smart transportation system use case, data will be generated from many sensors. These generated data will be analyzed and filtered in real time to get insight into the generated data. All generated data might not have any use. In some cases, it would not even be a good idea to store all generated data. As an example, if data is generated from a sensor each second, the mean value of data within a minute or within an hour may be stored depending on application requirements. Data can be trimmed in this way and a vast amount of storage space can be saved. In another case, if the difference among data values in some period of time is not that big, but might affect performance, then less numbers of reading within a minute can be taken. In such a way, it will be possible to filter a vast amount of generated data. Although the accuracy may not be 100%, application requirements might still be fulfilled to some extent. Data reconstruction is one of the components of this layer. This module takes care of faulty and incomplete data generated by the sensors. Similarly, if one or more sensors fail during operation, this component will reconstruct the data based on the data generation pattern to prevent interruption or any other application failure.

5) STORAGE LAYER

The storage module is responsible for storing data through storage virtualization. The data backup component ensures availability of data and mitigates the loss of data. In the storage virtualization concept, a pool of storage devices connected by a network acts as a single storage device, which is easier to manage and maintain. One of the key benefits of storage virtualization is to provide enterprise-class functionality using less-expensive storage or commodity hardware. Thus, the storage layer facilitates storage virtualization in order to minimize the complexity of the storage system. In a system, storage might fail at any point during system operation [67]. Therefore, it is crucial to backup important data to mitigate any unwanted situations. The data backup module in this layer takes care of periodic or customized data backup schemes.

6) RESOURCE MANAGEMENT LAYER

The components in this layer maintain the allocation of resources, and scheduling, and deal with energy saving issues. The reliability component maintains the reliability of application scheduling and resource allocation. Scalability ensures the scalability of Fog resources during peak hours where resource demand is high. The cloud deals with horizontal scalability while Fog computing aims to provide both horizontal and vertical scalability [9]. There are many distributed system resources for network, processing, and storage. This is a critical issue for distributed resources, which use application processing. Thus, resource allocation, deallocation, and reallocation will happen in which the resource allocation component manages and maintains resource allocation related issues. Another vital issue is that many applications will run in the Fog computing environment simultaneously. Hence, proper scheduling of these applications is required. The application scheduling component takes care of these applications based on various objectives. This layer also has energy saving components, which manage resources in an energy-efficient manner. Energy efficiency also positively affects the environment and helps minimize operational cost. Reliability components handle the requirement for the reliability of a system based on various reliability measures and metrics. Fog computing is a complex system that needs to take care of all IoT devices, Fog devices, and the cloud. Therefore, a device or connection might fail at any level, so reliability management is an important issue.

7) SECURITY LAYER

All security-related issues such as encryption of communications and secure data storage will be maintained by the components in this layer, which also preserve the privacy of Fog users. Fog computing is intended to be deployed as a form of utility computing like cloud computing. However,

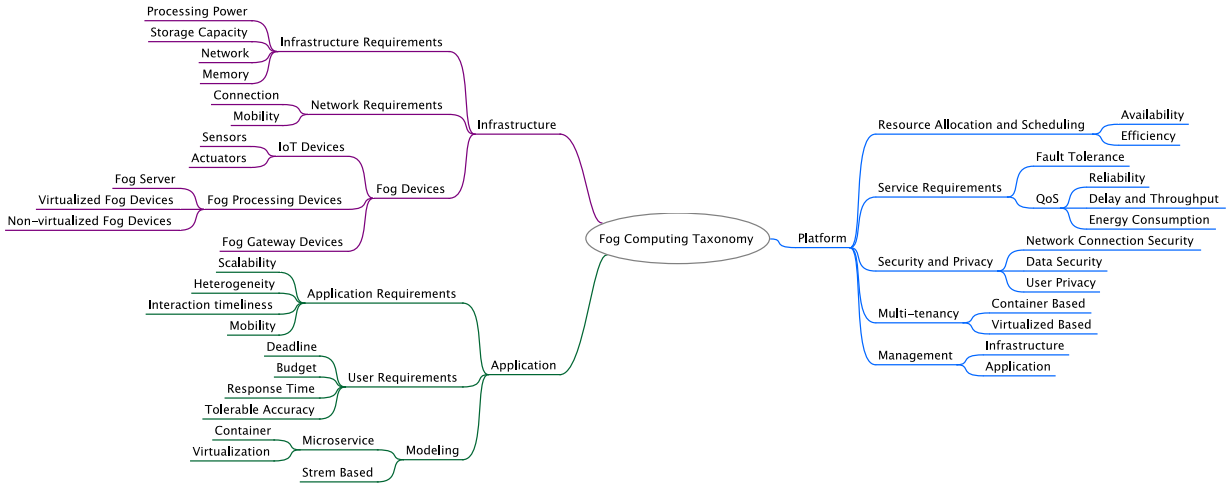

FIGURE 9. Taxonomy of Fog computing based on the requirements of infrastructure, platform, and applications.

in the cloud computing concept, the user connects to the cloud for services, but in the Fog computing concept the user will connect to the Fog infrastructure for the services while the Fog middleware will manage and maintain communications with the cloud. Hence, a user intending to connect to a service must be authorized by the provider. Therefore, the authentication component in the security layer processes authentication requests from users, so they can connect to the Fog computing service environment [27]. To maintain security, it is crucial to maintain encryption between communications, so that security breaches by outsiders will not occur. The encryption component encrypts all connections from and to IoT devices and to the cloud. Fog computing components are mostly connected via a wireless connection, so security concerns are crucial. Some services in a smart city or smart house privacy are also an issue because of the involvement of user-related data in these types of systems. The Fog computing paradigm should not disclose user information without their consent. In the current age, the majority of users normally accept the security policy of the provider without reading it. Thus, special consideration of privacy should be undertaken for such services that involve user-related critical information.

8) APPLICATION LAYER

Although the Fog was developed to serve IoT applications [58], several other applications based on Wireless Sensor Network (WSN) and CDN also support Fog computing. Any application that has latency-aware characteristics will be able to take advantage of Fog computing. This includes any type of utility-based service that could fit within Fog computing by providing better service quality and cost-effectiveness. For example, Augmented Reality-based applications should adopt Fog computing because of its nature. It is clear that Augmented Reality will transform the modern world in the near future. The needs of real-time processing for Augmented Reality applications can be addressed by Fog computing, which can maintain continuous improvement of Augmented Reality-related services.

VI. TAXONOMY OF FOG COMPUTING

The Fog computing taxonomy is presented in Figure 9. This taxonomy is derived by considering existing literature and the overall viewpoint on Fog computing. The proposed taxonomy focuses on the requirements perspective for infrastructure, platform, and application.

Firstly, by considering infrastructure, we identify infrastructure and network requirements, and the types of devices in a Fog computing environment. Secondly, for platform resource allocation and scheduling, security and privacy concern, service requirements, management, and multitenancy were determined. Finally, we defined application requirements, user requirements, and application modeling taxonomy for Fog computing. This taxonomy will help the research community and enterprises to gain better understanding and insight into the real-world deployment of Fog computing requirements, architecture, and devices. Figure 9 shows the taxonomy of Fog computing. A detailed description of each branch of the taxonomy is presented in this section.

A. INFRASTRUCTURE

Fog computing infrastructure requirements depend on the network, devices, and their requirements. All Fog devices, network devices, and gateways existing in the Fog environment that participates in computation are also part of the Fog infrastructure. Infrastructure denotes the physical components of the Fog environment.

1) INFRASTRUCTURE REQUIREMENTS

The many connected tiny devices are the primary elements in a Fog computing environment. These devices are located everywhere and help to connect all things around us. It is estimated that the world will see 50 billion handheld devices

by 2020. Beside these devices, a huge number of sensors and actuators will also be put in place. Therefore, a proper infrastructural facility is needed to support this vast computing environment [20]. An example of how the number of sensors is increasing day by day is given in The Economist report titled, ''Augmented Business'', which describes the implant sensors on cattle ears that could help to monitor their activity, health, and movements. This could help increase overall productivity. The implant of sensors affixed in one cow produces about 200 MB of information in a year. In another example, with sensor technology, Rolls-Royce is able to forecast when engines will more likely fail. From such a prediction, customers can plan engine changes. Heidelberger Druckmaschinen has huge printing presses equipped with more than 1,000 sensors. These are the examples of distinct uses of sensors in specific domains. However, this phenomenon will change completely when the distinct parts are connected to generate more efficient and effective decisions. Therefore, the Fog infrastructure must have the capability to provide physical resources for computation, networking, storage, and memory to achieve efficient Fog computing services.

2) NETWORK REQUIREMENTS

The network is one of the key bottlenecks in the Fog computing environment, where billions of connected devices generate and consume data at the edge of the network [68]. Most of the sensing and actuating devices require low bandwidth but a higher number of devices will be connected at the same time. Therefore, existing network connection technologies like LAN, MAN, WAN, or PAN need to be investigated further and amendments will be needed to cope with the Fog computing environment to facilitate countless IoT devices. Network operators are increasingly investing in new wireless access technology research because of the number of devices per user is increasing day by day. For instance, in the cellular mobile network, base stations have a limited number of link points [20]. As the number of things increase, these stations will need to support increasing numbers of devices. Fog devices must act as a router for neighboring IoT devices and as a primary processing unit for IoT application in the Fog environment. Each Fog device should have a resilient connectivity to the lower and upper layer devices. Mobile ad-hoc networks could act as a basis for the Fog network because of their mobility and lower cost feature [20]. Hence, connection and mobility are the main requirements for the Fog network.

3) FOG DEVICES

Fog computing is basically intended to support IoT-related technologies to perform processing at the edge level. Mine projects [69], [70] at the middle of the sea, airline fleets or a ship [70] can be equipped with a huge number of sensors, so it is impossible to send and store all generated data in real-time into the cloud. Some intermediate computation, processing, and services will be done by Fog computing devices. Thus, the Fog layer must have sensor management devices, Fog processing, and storage devices and Fog gateway devices. All of these devices will work collaboratively to manage and perform tasks in the Fog plane. Here, we discuss the devices that are needed for Fog computing deployment.

a: IoT DEVICES

IoT devices are the devices that have sensing and actuating capability. A sensor is able to sense the environment, while an actuator acts on it when necessary. One of the most common types of sensor in IoT devices is the temperature sensor. The temperature sensor has various functions depending on different domains such as at home, in factories, and in the agriculture field. This sensor is also used to sense the temperature of soil, water, and plants in order to take proper action needed to improve service outcome. Another type of useful sensor is the pressure sensor, used especially in agriculture, smart vehicles, and aircrafts. Sensors are also used to estimate the volume of water used by the agricultural sector for cultivation and other uses. Surprisingly, a huge percentage of this water is wasted due to leaky irrigation systems and inefficient use of fresh water. Efficient use of the pressure sensor will help solve this problem. The pressure sensor is able to determine the flow of water and reduce water waste. The pressure sensor is also used in smart vehicles to determine the forces acting on it, and in aircrafts to determine altitude.

Different groups of sensors are used for different IoT environments. For example, in healthcare, the most-used sensors are chemical, IR, pressure, and temperature sensors as well as other biosensors. On the other hand, in a smartphone, the most-used sensors are the gyroscope, GPS, and proximity sensors.

One of the applications of the proximity sensor is to determine the presence of ear to dim or turn off the phone backlight to improve battery efficiency. This sensor is also used to monitor parking space since it can determine the presence of an object without touching it. It can also be used in a wide temperature range and is not affected by color. Its detection process also is not effected by dirt, oil, or water. There are many other sensors out there that enable IoT, which include GPS sensors, water quality sensors, level sensors, chemical/gas sensors, smoke sensors, IR sensors, humidity sensors, sound and vibration sensors, motion sensors, acceleration sensors, and machine vision sensors. There are five main types of actuators-magnetic or thermal, electrical, hydraulic, pneumatic, and mechanical. The actuator has a controlling or moving mechanism, a motor, which acts on various inputs.

The raw application data comes from various sensors like speed sensors, cameras, temperature sensors, vehicle monitoring sensors, or GPS sensors. A typical sensor generates 10 data samples every second [71]. Sensors convert environmental variables such as smoke, heat, light, temperature, humidity, sound, and so on into electrical signals. These sensors are varied and can be micro-electro-mechanical systems (MEMS)-based, CMOS-based or LED sensors. Communication among sensors could be done by ZigBee,

Bluetooth, Z-Wave or 6LoWPAN standards for short distance communication [72]. There is a necessity for communication among sensors in some cases where one sensing output is dependent on other collective sensor outputs. These sensors will be connected to Fog devices through wireless connections. However, Fog devices collect and process data based on application requirements.

Some example of research works based on sensors can be improved by taking advantage of Fog computing. Aziz *et al.* [73] proposed a real-time health monitoring system using particular sensors in which the proposed architecture was based on GSM and GPS technologies. The system specifically monitors the body temperature and blood pressure of patients. The study used an Arduino microcontroller, dfrobot GPS/GPRS/GSM module v3.0.3, a heartbeat pulse sensor, and a lilypad temperature sensor for hardware implementation. In another study, a web-based application was developed for doctors and nurses with SMS functionality, which will be used as an emergency case. The system is able to generate GPS location, body temperature, and blood pressure. Butt *et al.* [74] investigated wearable technology such as SensHand, Gloves-based system, electromyography-based and hybrid systems, leap motion, and smartwatches. The development of these kinds of technology must be integrated with the smart home system and Fog-like architecture in order to deal with emergency situations.

Some devices such as the smartphone can be considered as both an IoT and Fog device. In the same way, if some sensors and actuators were installed in the Raspberry Pi, the device could also act as both an IoT and Fog device.

b: FOG PROCESSING DEVICES

Any device that has computing capability, storage, and network connectivity can act as a Fog processing device. It could be a network controller, switch, router, server, or a video surveillance camera. A Cisco 800 series router can be used as a Fog device where the IoT application can be run on the device and the device supporting Cisco IOx. To date, only Cisco 800 series routers are supporting IOx with Linux kernel with virtualization support [44]. Most of these devices have a 266-400 MHz MPC8272 processor with 16 KB Cache, 64 MB random access memory and 20 MB processor board flash memory. The user can host an application on these routers. These routers have two cores-Cisco IOS runs on one core and another core is used for running IOx services.

Another type of Fog processing device is the Fog server. A Fog server can control several Fog devices in a specific domain. Cisco offers two flavors of Fog computing server deployment. One is the Cisco Fog Director, which can be deployed on any type of server with Cisco-recommended server specifications [66]. Another example of a Fog device manufactured by Cisco is the Fog data services, which are specifically designed for IoT [65]. However, Cisco Fog data services can only be deployed on Cisco UCS E and C Series Servers. Both will act as Fog servers; however, Cisco Fog data services are especially designed for an IoT environment. However, various organizations and bodies need to work beyond the proprietary solutions for fast technological advancement and technology adoption with a limited budget.

Fog devices and Fog servers should be deployed in such a way that any type of network management device with storage and processing capability can act as a Fog device. Similarly, the usual type of server must be able to act as a Fog server. This could be an ordinary PC since Fog is not dedicated to performing very complex processing. However, further investigation is necessary to explore the minimum system requirement for a device that can act as a Fog device or Fog server. Connectivity between Fog devices and Fog servers will be via Ethernet or wireless or a serial connection in some cases. As an example, Cisco UCS E and C Series Servers, which are generally used as Fog servers, are connected to the network via Ethernet. On the other hand, Cisco 800 series routers are connected via serial ports that support Fog computation.

c: GATEWAY DEVICES FOR FOG

Many hardware boards are currently available in the market including Arduino Yun, Intel Edison, Raspberry Pi, Beaglebone Black, Arduino + Shields, Netduino, Tessel 2, and so on. These boards are currently used as IoT and gateway devices and can also be used as Fog gateway devices and as Fog devices. These boards have a built-in processor, wired and wireless adapter, and a USB port. Fog computing supports device heterogeneity, where a gateway could also be a part of the Fog computing environment. Constant *et al.* [75] developed a Fog gateway using Intel Edison and Raspberry Pi. Their proposed Fog gateway integrated the data conditioning process, smart analytics, intelligent filtering, and transfer to the cloud, which needs long-term storage and temporal variability monitoring.

The IoT gateway supports various data types and communication protocols between devices and sensors. It also unifies the data format from various sensors. Current IoT gateways provide a solution for communication and do not support fully automatic configurations of newly added IoT devices [76]. Guoqiang *et al.* [77] proposed a smart IoT gateway with three key benefits. The proposed gateway has a unified external interface and pluggable architecture. It has a flexible protocol to translate various sensor data into a uniform format. The study designed a customized device with a Samsung S5PV210 mobile application processor as its gateway. However, this gateway did not have any fault tolerance or security features.

B. PLATFORM

The platform manages applications and infrastructure in the Fog environment. It takes care of resource allocation, scheduling, fault tolerance, multi-tenancy, security, and privacy in Fog computing. Based on the taxonomy of the Fog, we discuss the requirements of the platform for Fog computing in this section.

1) RESOURCE ALLOCATION AND SCHEDULING

Heterogeneous devices are the main challenges in developing proper resource allocation and scheduling in the Fog. If we wanted to use the computation power of idle devices, we need to schedule tasks on these devices efficiently. Otherwise, IoT application processing in the Fog will face complex issues, which will hinder the fulfillment of the latency awareness goal. Two of the key requirements for resource allocation and scheduling are availability and efficiency. Resources in the Fog are not dedicated, and thus availability should be ensured. On the other hand, lack of efficient resource allocation and scheduling might lead to unwanted delays in the overall processing.

2) SERVICE REQUIREMENTS

Fog services can be defined as single or multiple user requests, where the user will constantly be updated of the outcome of the service until he or she has a subscription to that service. This means that the service outcome will not be fixed and will keep changing until the end of the service. The Fog device and Fog server perform the intermediate processing, which occurs in between user request and service output. The Fog server may communicate with the cloud for processing and information retrieval when necessary. For instance, if we considered selecting the best path based on real-time traffic in a smart transportation system, the Fog service will keep updating on the best path until the end of the journey. In this case, we need to take into account mitigation of fault, service quality, network latency, and power consumption in order to maintain the standard of the service.

a: FAULT TOLERANCE

Fault tolerance allows a system to keep performing even when a part of the system has failed. This failure might be software failure, hardware failure, or network failure. The solution for fault tolerance will result in a fully operational system where the system will continue its operation with a lower capability instead of shutting down totally [78]. Fault tolerance is mostly investigated in the cloud [79]–[89]. However, it is necessary to investigate fault tolerance in the Fog as well. Although many research works have addressed the need to explore fault tolerance issues [5], [9], [64] in Fog computing, none have actually investigated the issue. We discuss in more detail the issue of fault tolerance in Section VIII-B.

b: QUALITY OF SERVICE (QoS)

QoS is an important service requirement for Fog computing, which is based on reliability, network delay, throughput, and energy consumption. Besides these, resource management, power-consumption model, scheduling policy, and power failure handling are also important to ensure QoS. If some sensors fail for any reason, the accuracy of the outcome or action could be affected. Fog is intended to work with latency-sensitive systems; hence, it should maintain high reliability with a strict QoS assurance. Otherwise, the latency awareness criteria will not be fulfilled. Madsen *et al.* [90] suggested that the availability of different methodologies and algorithms work with the reliability of network connectivity and information, to ensure accuracy, which is crucial for building Fog computing-based projects.

3) SECURITY AND PRIVACY

In this technological era, people are inevitably sharing personal information when using different applications and web services. Our personal information is no longer personal; it now belongs to many tech giants because we are using their free services on a regular basis [91]. A simple example is that if anyone used an Android phone without any security settings, the built-in Android OS will automatically run GPS and map services, for which it can collect all location-related activities about the user. Therefore, information about when and which country a user has visited, where a user has dined in, which route a user uses for going to the office, home, and so on will be made available to these companies. However, these tech giants might argue that they do not disclose our data to others, as they can only see our data in our timeline only. However, a recent Facebook incident fails to convince us of the honesty of these tech giants [92].

The Fog computing paradigm is completely distributed and not intended to be centrally managed most of the time. Sensitive data might be processed in an intermediate device when the application does not have full control of the device. On the other hand, the user will not have full control over the Fog applications. Users will require more protective and innovative ways to retain their privacy and protect it from any potential and very harmful entities [20]. Similarly, Fog application providers also need to develop security to protect their application from unwanted data theft.

Three different types of security need to be ensured: network connection security, data security, and user privacy. Network connection security and data security are applicable from both the user and provider perspectives. Moreover, user privacy is also important because Fog processing is carried out on user data in most cases.

4) MULTI-TENANCY

Multiple tenants for the same services with an isolated runtime for each tenant are referred to as multi-tenancy service. Multi-tenancy is important for Fog because of the limited resources in a Fog environment. By enabling multi-tenancy, one instance will run in a Fog device and will serve multiple tenants (users). Multi-tenancy could be container-based or could be the usual virtualization-based. Container-based virtualization is a more lightweight and powerful virtualization solution, which the Fog can adopt, to provide the fastest processing solution. Container-based virtualization does not need to emulate the operating system to facilitate virtualization; thus, it will be easier to manage and migrate. Multi-tenancy is a requirement for the platform, and needs to be defined before deployment. Multi-tenancy may

incur performance degradation and security issues [29]; thus, adequate and secure isolation is needed.

5) MANAGEMENT

The management of the Fog can be centralized or decentralized. Since the devices in a Fog environment belong to different domains, centralized management is not always possible. Alternatively, processing of IoT applications will be done in different Fog clusters, so management will follow a distributed nature in this case. In summary, the management of the platform in a Fog must be defined. In the case of decentralized management, similar processes must be deployed for different Fog devices to handle management issues.

C. APPLICATION

Applications have to fulfill certain requirements to execute in a Fog environment. Here, we discuss the features required by the applications for execution.

1) APPLICATION REQUIREMENTS

a: SCALABILITY

The number of IoT devices are increasing very swiftly day by day all over the world, which raises a new issue of scalability. Thus, we need to deal with the scaling of devices and services in the Fog computing environment. Dependency on cloud computing has been observed for IoT application processing by many research works, where trillions of IoT devices are involved, such as that of Li *et al.* [93]. However, implementation of the whole application in the cloud in such an environment where IoT devices are generating a huge amount of data is neither feasible nor efficient. IoT devices are not only stationary but also mobile in most cases. Hence, maintaining frequently changing device states and availability in the cloud is not an easy task. Also, with the growing number of IoT devices, it would be more critical for IoT applications to query and select IoT devices [59]. The Fog computing system must be an autonomous system where application execution by the participating device will be done automatically including scalability.

b: HETEROGENEITY

For any IoT system, the heterogeneous device is a fundamental characteristic where device heterogeneity co-exists at any level in the Fog computing paradigm. Abstraction of device complexity is also required to some extent. Device heterogeneity does not only refer to the diversity of services and protocols, but also the assortment of horizontal and vertical levels of the Fog architecture [59]. To address this heterogeneity, Giang *et al.* [59], classified three types of Fog devices: compute, input/output (IO), and edge nodes. Edge nodes are the sensors and the actuators, IO nodes are the resource-limited devices mostly responsible for brokering communications, and computing nodes offer computing facilities. Of the three types of nodes, IO and compute nodes are mostly dynamic and customizable or programmable as required. It is possible to implement all three nodes in a single device based on its capability and design goal. The smart gateway is an example of such implementation. In order to use the capability of various types of devices in an IoT environment, it is obvious that the application must be designed in such a way that it might be able to perform its task execution on multiple devices regardless of its capacity and location. More precisely, the application should able to use maximum available computation resources through middleware.

c: INTERACTION TIMELINESS

The perception-action (PA) cycle is the basic function of a nervous system, which maintains circular flow between sensory organisms and its actions towards the functionality of that sensing. The PA cycle is also a characteristic of IoT applications, where the cloud and Fog infrastructure satisfies timeliness requirements and application logic for communication. Giang *et al.* [59] identified four interaction models for the PA cycle in a Fog environment. Examples of these models are: (i) in a local network, communication between devices, which is considered as an immediate cycle action, (ii) interaction with the cloud from a device of a local network, which is generally for time-insensitive actions, (iii) an interaction generated by the cloud to a device in a local network, which requires semi-immediate actions, and (iv) communication among IoT-related applications in the cloud. However, their work considered the role of the Fog server, which manages and maintains several Fog devices in a specific cluster. On the other hand, PA interaction can be divided into immediate, semi-immediate, and delayed action to leverage IoT application requirements more efficiently. Delayed action can be performed on the type of processing that does not have any timeliness issues and could be processed by the cloud infrastructure.

d: MOBILITY

Device mobility is a natural probability and is one of the key requirements for implementing an IoT platform [59]. From the Fog perspective, it is not only the edge devices that will be mobile but also computing and storage devices in the Fog layer. Managing mobile devices in two different planes and syncing them with each other is challenging. To ensure resource availability and successful task completion, task distribution, duplication, and migration is required. This mechanism is already considered in the cloud but there is a need to reinvestigate them by considering mobility [94].

2) USER REQUIREMENTS

User requirements can be changed by various constraints. First of all, a user may want to complete the submitted task within a specific time binding, which is referred to as the deadline. Secondly, the user may set some constraints for the budget. Thirdly, in the case of some users, they may not care about the budget but the response time is of utmost importance. Fourthly, some users may want tolerable accuracy. This means that the user may not seek

accurate results but rather fast results that could be provided with some reasonable errors. Aazam and Huh [54] suggested that pre-allocation and prediction of resources rely on user behavior and the probability of future utilization of resources. Dastjerdi *et al.* [4], [95] stated that edge devices perform optimization by considering user behavior and network condition.

3) APPLICATION MODELING

Two types of application modeling are possible by considering the requirements of applications in the Fog. Most IoT devices generate tuples periodically, which can be considered as a stream. These streams need to be processed in real time to get accurate results. Alternatively, the application that does the processing based on previously stored sets of data could include microservice-based applications. The advantage of microservice is that it can bind all functionality and required libraries in a single service, which can run above the microservice controller without dependency. Hence, application modeling in Fog could be stream-based or microservice-based.

VII. DIMENSION OF FOG COMPUTING-BASED APPLICATIONS

Several applications require a Fog computing infrastructure to provide smooth services. These include smart transportation systems, Augmented and Virtual reality, healthcare, video streaming, smart homes, and smart cities. Requirements of platform and applications are also needed in order to provide services. In this section, we discuss some research works, which specifically address the application of Fog computing. We evaluate each work based on their contribution on the Fog infrastructure, platform, and applications as defined in our taxonomy. It is obvious that all three kinds of services are interrelated. However, each researcher only focused on one or more of these aspects. Mapping related works with our proposed taxonomy will help in finding the research gaps in Fog computing applications.

A. SMART TRANSPORTATION SYSTEM

Several research works have been carried out on smart transportation systems that use Fog computing. In this section we discuss a few works that have been done on the Fog-based smart transportation system and then identify key issues that need to be addressed.

Truong *et al.* [96] pproposed a Vehicular Ad-hoc Network (VANETs) architecture called Fog Software Defined Networking (FSDN), which combines SDN and Fog together to provide a better solution. As SDN has programmability, flexibility, global knowledge, and scalability features and Fog has location awareness and time sensitivity, the combination of these two will leverage on the key challenges in VANETs. The proposed system is able to augment communication among vehicles, infrastructure, and base stations via centralized control, besides reducing latency and optimizing the utilization of resources. However, the central SDN controller of the proposed system is where the bottleneck of the proposed system occurs. The system is focused on infrastructure and network requirements. The Fog controller is used for service implementation. The work did not focus on platform and application requirements.

Investigation of VANETs in Fog has also been done in Giang *et al.* [97], where they explored how smart transportation applications (VANETs) are developed using the Fog Computing approach. Driving vehicles in an urban area requires immediate decision on various activities such as route changing, lane change, slowing down speed, looking at obstacles, and so on. Hence, applications need to gather all related details to act on these activities. The authors discussed Fog-based smart transportation application requirements such as programming abstraction and application models. The work explored application modeling but not other application aspects nor infrastructure or platform.

B. VEHICLES AS FOG INFRASTRUCTURE

Hou *et al.* [98] proposed the idea of Vehicular Fog Computing (VFC), which will use the vehicle as an infrastructure for computation and communication. The VFC architecture utilizes vehicle computation resources by providing service to the edge devices located near them. It will aggregate abundant resources of each moving car by which service quality can be enhanced. Using quantitative analysis on different scenarios, they discovered an interesting relationship among connectivity, mobility, communication capability, and parking behavior. These four characteristics help us understand resource utilization of vehicle resources, which will help achieve better utilization of unused resources.

C. AUGMENTED AND VIRTUAL REALITY

Augmented Reality applications are extremely time sensitive; a small delay can lead to serious errors in user experience. Thus, Fog computing-based solutions will have great potential in this domain [4]. These statements are also applicable for connected Virtual Reality (VR) or VR-based games. Zao *et al.* [99] proposed an augmented brain computer interaction game, which utilized the Fog and cloud infrastructure. The Fog performed real-time analysis such as signal processing that needs to classify the brain state and other analyses such as model classification updated from the cloud. However, their work only focused on the Fog infrastructure but neglected most aspects regarding platform and application.

D. HEALTHCARE

The Fog computing approach also enables real-time sensor-based healthcare services. Rahmani *et al.* [100] proposed a Fog-assisted system architecture for the healthcare system. A smart e-health gateway is the key component of this architecture, which will process the generated data from the sensors and generate an Early Warning Score (EWS) to notify for any medical emergency. They considered many aspects of our taxonomy; but it is necessary to investigate each aspect extensively, which this study did not. Another Fog-based

TABLE 4. Evaluation of existing works on Fog applications.

Author & Year	Application Type	Infrastructure			Platform					Application					
										Application requirements				User Requirements	Application Modeling
		Infrastructure requirements	Network requirements	Fog Devices	Resource Allocation and Scheduling	Service requirements	Security and privacy	Multi-tenancy	Management	Scalability	Heterogeneity	Interaction timeliness	Mobility		
Truong et al. [96]	Smart Transportation System	✓	✓	✓	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗
Giang et al. [97]	Smart Transportation System	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✓
Hou et al. [98]	Vehicles as infrastructure	✓	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗
Zao et al. [99]	Augmented and virtual reality	✓	✗	✓	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✓
Rahmani et al. [100]	Healthcare	✓	✗	✓	✗	✗	✗	✗	✓	✓	✗	✗	✓	✗	✓
Mahmud et al. [101]	Healthcare	✓	✓	✓	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✓
Giordano et al. [102]	Smart City	✓	✗	✓	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✓

healthcare architecture was proposed by Mahmud *et al.* [26]. Their work mainly focused on network delay, power consumption, and communication optimization in Fog-based healthcare service. However, platform, application, and user requirements were not investigated.

E. SMART CITY

Smart city-related applications need to process sensor data on a real-time basis, where Fog computing can play a major role. Giordano *et al.* [101] proposed a Rainbow architecture, which supports various applications in a smart city. The proposed Rainbow framework evaluated three smart city applications including noise pollution mapping, urban drainage networks, and smart street. The work proposed a distributed agent-based approach in the intermediate layer in between the physical infrastructure and cloud. However, the work did not focused on application and platform aspects except for application modeling.

Table 4 shows a summary of the above-discussed Fog-based applications mapped to our proposed taxonomy. In summary, it can be concluded that most of the works have focused on infrastructure and application modeling. There is a research gap on application- and platform-related aspects, which need to be explored further.

VIII. STATE-OF-THE-ART FOG COMPUTING

In this section, we focus on some existing research works on Fog computing. We discuss research works from four different research areas of Fog computing. These areas are resource allocation and scheduling, failure handling, simulation tools, and microservices.

A. RESOURCE ALLOCATION AND SCHEDULING IN FOG COMPUTING

Fog computing is fast evolving and growing rapidly due to its edge-level computation and heterogeneous nature. In this section, we present several research works, which have been done in the past couple of years. We also summarize the presented research works with a comparative discussion to address research gaps in this area. Most of the reviewed works are related to resource allocation and scheduling in the cloud and Fog environment. However, some works have only been done in the Fog environment.

1) RESOURCE ALLOCATION AND SCHEDULING FOR FOG-CLOUD ENVIRONMENT

Alsaar *et al.* [102] proposed resource allocation methods for a collaborative platform composed of Fog and cloud paradigms. Their proposed algorithm is grounded on

linearized decision tree rules by considering three different conditions for managing user request and for balancing workload. The conditions are VM capacity, completion time, and service size. Each condition has two branches: the VM capacity branches out to enough or not enough; the completion time consists of now or later, and the service size is divided into small or large. In some cases, this includes services in the queue, which will be represented with yes or no. They utilized 1/m/m/1, with (1)/ representing cloud broker, /(m) for many paths, /(m) for many Fog brokers, and /(1) for IoT device users. Using this method, the total overhead for big data processing in the system was reduced. In their work, the availability of cloud servers and the Fog was guaranteed and a fast response time to satisfy QoS was achieved. The SLA for users was also different, where shared and reserved resource was provided. However, availability and QoS were not studied extensively.

Deng *et al.* [103] presented a framework for workload allocation in the cloud and Fog environment to examine power consumption-delay trade-off issues. They defined the workload allocation problem into primary and sub-problems, which can be solved via related sub-systems. They employed a Hungarian algorithm and Generalized Benders Decomposition (GBD) algorithm to solve the problem. Numerical and simulation results were presented to prove that the Fog is a complement to the cloud. However, the complex nature of workload and resource was not studied in their work.

Brogi *et al.* [104] prototyped a tool known as 'FogTorch$\prod$' which is capable of fulfilling hardware, software, and QoS requirements before deploying a composite application in the Fog infrastructure. The proposed tool manipulates Monte Carlo simulations and only considers communication link QoS. Resource consumption and QoS assurance terms were undertaken for classifying the eligibility of deployments. The proposed algorithm was based on the preprocessing phase and backtracking search phase. To find eligible deployment, the preprocessing used input from results derived by the backtracking search algorithm. However, availability and latency are more important in the Fog environment compared to resource consumption and communication links.

In order to ensure efficient use of resources and network infrastructure in the Fog and cloud environment, Taneja and Davy [105] proposed a Module Mapping Algorithm, which efficiently deploys IoT Application Modules in the composite Fog-Cloud Infrastructure. They employed lower-bound searches and compared function algorithms to find an eligible network node in the Fog and cloud. The Module Mapping algorithm returned a map with nodes, which are appropriate for completing the computation operation. If the application requires faster processing, the application will be deployed close to the source device. However, the work considered CPU, RAM, and bandwidth to find the best resources. In such a case, the cloud resource will always be the best resource, so it will be necessary to consider other parameters such as response time and availability of the specified resources.

Yin *et al.* [52] studied a Fog-assisted big data streaming scenario, where Fog devices are responsible for preprocessing raw data for applications hosted in the cloud using the unused resources of Fog devices. In their work, the software-defined network (SDN) controller continuously adjusted the volume of data to be sent to the Fog device for pre-processing. The collaborative computation problem was defined as a social welfare maximization problem and a hybrid alternating direction method of multiplier (H-ADMM) algorithm was proposed to minimize computation burden via the dynamic distribution of Fog devices, cloud, and SDN using message exchanging. The formulation of social welfare maximization problem determined the size of data that will be assigned to a Fog device. During the formulation, loss of information value by preprocessing and the operation cost of the Fog and cloud were considered. The work completely depended on the cloud for post-processing, but pre- and post-processing could have been done in the Fog to support time-sensitive real-time applications.

Aazam *et al.* [106] proposed a dynamic resource estimation algorithm by integrating the historical record of cloud service customer (CSC) in a Fog environment based on the relinquish probability. The minimum relinquish probability value is 0.1 and this value will be increased based on the history of the user. However, for fair resource estimation, the relinquish probability will be 0.3 for new customers. For existing and returning customers, the characteristics of the customer are known, so the probability value can be calculated easily. In this way, resource underutilization could be minimized and the chances of profit loss will be low.

2) RESOURCE ALLOCATION AND SCHEDULING FOR A FOG ENVIRONMENT

A resource allocation strategy based on priced timed Petri nets (PTPNs) was proposed by Ni *et al.* [107] for Fog computing. The main idea of this work is that the user can choose the satisfying resources autonomously from a pre-allocated resource group. With credibility evaluation for both users and Fog infrastructure, their proposed strategy comprehensively considers the cost for time and price to complete the tasks. The user that has a high credit limit will be able to allocate highly reliable resources to complete their tasks. Due to the dynamic nature of creditability of users and resources, there will be some deviation in calculating them properly. To maintain QoS, the resources will be ordered according to their processing capacity and divided into several groups. Moreover, users with similar credibility will be assigned to several groups.

Pooranian *et al.* [108] proposed a simple algorithm to find an optimal solution for resource allocation. They considered the problem as a bin packing penalty aware problem where servers are bins and VMs are the pack. Based on idle energy, maximum frequency, and maximum energy, each server will be palatalized and rewarded. The method will calculate how many VMs could be allocated in t time slot on a server. The VMs will be served based on their frequency

and time limitations. As a consequence of penalty, a server will be punished in the form of being banned from use for a few iterations. Once the server passes the iteration freeze, it will return to the stream to perform further computation. The penalty and reward methods are applied to minimize exponentially increasing energy consumption.

Sun and Zhang [63] proposed a crowd-funding algorithm for a Fog environment, integrating idle resources in the local network. An incentive mechanism was used to encourage resource owners to participate in the computation and enthusiastically perform their tasks. Through the comprehensive reward and punishment mechanism, it is ensured that the participant will positively perform the tasks. This work is similar to the above-described literature proposed by Pooranian *et al.* [108]. However, in this case, the reward and punishment go to the participant rather than the physical server.

3) SUMMARY OF RESOURCE ALLOCATION AND SCHEDULING IN FOG

Based on the related research on resource allocation and scheduling in the Fog, a summary is presented in Table 5. From this table, we can see that most of the researchers have focused on resource allocation in the Fog. More research works are therefore required to investigate resource sharing and workload allocation. Also, further investigation is needed to address energy-efficiency, load balancing, SLA, and QoS in the Fog. We identified two major issues in Fog computing research. Firstly, researchers tend to use a synthetic workload to validate their methods and algorithms. Secondly, most of the researchers used cloud-based simulations, which are not that convincing because the Fog is more heterogeneous and dynamic in nature. Thus, further investigation into workload generation and simulations in the Fog need to be undertaken.

B. FAULT TOLERANCE IN FOG COMPUTING

The Fog computing paradigm is a highly distributed heterogeneous platform where the probability of device failure is very high compared to the cloud. Since the Fog is evolving, no study has yet been done on fault tolerance in Fog computing. However, fault tolerance has been mostly studied in the cloud computing paradigm.

Often, fault tolerance is measured by availability. In the cloud, faults are handled by proactive fault tolerance and reactive fault tolerance techniques at either the workflow level or task level. Reactive fault tolerance techniques are used to reduce the impact of failures on a system when the failures have actually occurred. Techniques based on this policy are job migration, checkpoint/restart, replication, rollback and recovery, task resubmission, user-defined exception handling, and workflow rescue. Proactive fault tolerance predicts the faults pro-actively and replaces the suspected components with other working components; thus, avoiding recovery from faults and errors. Proactive Fault Tolerance uses self-healing, preemptive migration, and software rejuvenation, which are the few proactive fault tolerance techniques in the cloud.

According to Sharma *et al.* [110], the causes of failure in the cloud varies, and include software and hardware failure, service failure, overflow failure, power outage, outdated systems, network failure, cyber attacks, and human errors. It is crucial to handle faults in Fog computing for which the fault needs to be considered at every step, not only for processing but also for the transmit-and-receive process [111]. In this section, we discuss some existing research works on fault tolerance in cloud computing. We specifically focus on resource and task failure mechanisms. Then, we summarize the existing works and present a research direction for failure handling in Fog computing.

Jiang and Hsu [112] proposed a two-level standby design for handling server failure in the cloud system. In their proposed system, cold and warm standby of the system is made available. Once any server fails, the warm standby system will replace the failed server and the failed server will be sent to the repair house. After repairing, the system will be placed in the cold standby group. The systems in the cold standby group are in a completely switched off mode. The work proposed a model to determine the necessary number of cold and warm standby systems in the cloud. However, this type of hardware failure handling is not suitable for the Fog because most of the time Fog computing devices will not be under the property of the Fog provider. Hence, task migration is the best solution for hardware failure and this should be reactive in most cases, except where the Fog device belongs to the provider. Latiff *et al.* [113] proposed a cloud-scheduling scheme based on a check-pointed league championship algorithm. They employed a task migration method for independent task execution failure. In their proposed method, the system state will be saved periodically by check-pointing, so the task need not start from the beginning once it fails. When the task fails, it will be assigned to an underloaded VM and the league championship algorithm will be employed to schedule the failed tasks.

Wu *et al.* [114] proposed a fault tolerance technique using migration to the cloud. The failure handling method is proactive, which always monitors the host and continuously tries to predict the chances of failure. If the prediction becomes true, the system will look up other available resources and then migration will be performed. The proposed method will monitor CPU temperature, memory usage, and CPU fan speed, etc. To employ such a technique in Fog computing, further investigation is needed because the types of device in Fog are diverse.

A combined method of check-pointing and migration-based proactive failure handling was proposed by Egwutuoha *et al.* [115] for HPC and cloud. In the proposed method, the authors used a Lm-sensors open source software tool for computer health monitoring. From the monitoring data, they defined rule-based monitoring depending on temperature, fan speed, voltage, and processor utilization to predict failure. The rules are denoted as 1, 2, and 3,

TABLE 5. Summary of resource allocation and scheduling research in Fog.

Author & year	Proposed Algorithm	Infrastructure	Method applied	Evaluation Process	Workload	Method proposed for				Other Considerations			
						Resource Allocation	Resource Provisioning	Resource Sharing	Workload allocation	Energy Efficiency	Load balancing	SLA Parameters	QoS parameters
Alsaffar et al. [103]	Resource Allocation and Data Distribution	Fog / Cloud	Utilized 1/m/m/1 model where (1) refers to cloud broker, (m) refers to many paths, (m) refers to many Fog brokers in Fog environments, and (1) refers to IoT devices users.	Simulation (CloudSim)	Synthetic	Yes	Yes	Yes	No	No	No	Response time, target time	Server availability, fast response time
Deng et al. [104]	Generalized Benders Decomposi tion (GBD) algorithm and Hungarian Algorithm	Fog/ Cloud	Define optimal workload allocation among cloud and Fog which minimizes power consumption with a defined service delay. Proposed technique solved workload allocation problem in cloud and Fog.	Simulation (MATLAB) and Numerical	Synthetic	No	No	No	Yes	Yes	No	NA	NA
Ni et al. [109]	Dynamic resource allocation strategy	Fog	Ordering the resources into several groups based on their processing capabilities.	Testbed	Synthetic (Random)	Yes	No	No	No	No	No	NA	Price, completion time
Brogi et al. [105]	Backtracking search algorithm	Fog/ Cloud	Define the deployments of complex applications to Fog infrastructures, which fulfil hardware, software and service quality requirements.	Simulation	Yes	No	No	No	No	No	NA	Latency, bandwidth	
Taneja et al. [106]	ModuleMapping and LowerBound Algorithm	Fog/ Cloud	Efficient application deployment in Fog-Cloud environment for IoT based applications by employing module mapping algorithm for efficient utilization of network infrastructure and resources.	Simulation (iFogSim)	Synthetic	Yes	No	No	No	Yes	No	NA	Latency
Yin et al. [107]	Distributed Resource Sharing Scheme	Fog/ Cloud	Grounded on an algorithm known as Hybrid Alternating Direction Method of Multipliers (H-ADMM) algorithm	Numerical	NA	No	No	Yes	No	No	No	No	No
Pooranian et al. [110]	Energy- aware algorithm for Fog data center	Fog	Provide reward and penalty to each server depends on their energy characteristics such as maximum frequency, maximum energy and idle energy	Numerical and Simulation (iFogSim)	Worldcup98 workload [111]	Yes	No	No	No	Yes	Yes	No	Latency
Sun et al. [63]	Crowd-funding algorithm	Fog	A resource-sharing model grounded on the repeated game theory in Fog computing paradigm	Hadoop cluster	Synthetic (Jmeter)	No	No	Yes	No	No	No	SLA violation	No
Aazam et al. [108]	Dynamic resource estimation	Fog/ Cloud	Relinquish probabilities of the customers by considering resource utilization and the service quality experienced	Simulation (CloudSim)	Real IoT traces	Yes	No	No	No	No	No	NA	NA

representing normal, warning, and critical state, respectively. They employed three different policies for migration. The first depends on the necessity lease additional node. The second removes the node, which is unhealthy based on the state. In the third, the critical state publishes to the head node. Finally, the system administrator is notified for further action. This type of approach might increase the overhead in the Fog; however, further exploration is essential. A recent study shows that proactive fault tolerance is the best solution for the cloud compared to redundant solutions [88]. However, failure prediction accuracy is the key factor for these kinds of solutions. Their work considered software, hardware, and unstable behavior to predict the failure of the infrastructure. More specifically, they defined failure based on an error formula $err = (ActualTime - PredictedTime)/ActualTime \times 100\%$, which was derived from [118] and [119]. A combination of the proactive and reactive method was applied by Gao *et al.* [118] to handle task failure in the cloud environment. The crash detection method and replication factor were proposed in this work to handle failures. Table 6 shows a summary of the investigated literature on fault tolerance in the cloud.

Because of its unstable nature of failure and heterogeneous characteristics, the hybrid failure handling method is more appropriate for the Fog computing environment.

C. SIMULATION TOOLS FOR FOG COMPUTING

Simulation and modeling in Fog computing are still in their infancy. However, a few research works have been done on Fog computing simulation, which are focused on some specific aspect of Fog computing. Aazam and Huh [54] focused on resource prediction and pricing in Fog computing. The Proposed Fog-based resource management model is able to estimate the required resources based on the probability of user behavior of future resource use. Validation and performance evaluation was done using simulation. However, they did not consider service heterogeneity, QoS, or device mobility factors. Another work proposed by Dastjerdi *et al.* [4] focused on dag of the query for incident detection in a smart city use case. Both of these works used CloudSim [119] to validate their method along with an experimental evaluation. The first toolkit for Fog simulation was developed by Gupta *et al.* [120], known as iFogSim. The toolkit is used for the simulation and modeling of IoT resource management techniques in the Fog and edge computing paradigms. The most challenging problem is the design of resource management techniques, which determine analytic application distribution among edge devices, which will improve the throughput and reduce latency. The proposed simulator is capable of measuring the impact of resource management techniques in terms of network congestion, latency, cost, and energy consumption. The simulator was validated using two use cases and the authors also proposed a Fog computing environment architecture.

Challenges in Fog computing deployment are include incorporating Fog with Emerging Technologies such as 5G Technologies [121], Network Function Virtualization (NFV), and Software-defined Networking (SDN). In this case, a simulator with container, SDN, and NFV support is crucial.

Table 7 presents the key features of these two simulators that are mostly used by various researchers for Fog computing simulation. These two simulation tools did not focus on network parameters such as bandwidth distribution of the link and round-trip delay of the various media. These two parameters heavily affect the simulation results where minimization of latency is the key goal in a Fog computing environment. Secondly, both tools did not consider container-based virtualization. In a Fog computing environment, there are many devices that will participate in computation, where hypervisor-based virtualization is nearly impossible to implement due to the lower memory and processing power of these devices.

D. FOG-BASED MICRO SERVICES

A microservice is an independent process and Software-Oriented Architecture (SOA) that interacts by message passing. The SOA of microservice does not hinder or favor any specific programming model. It provides design and implementation guidelines for distributed applications to partition each component independently. Each of the components addresses a specific functionality. The functionality of the components can be accessed by message passing and is possible to implement in any mainstream programming language internally. In this way, this principle helps developers and project managers to develop each module independently and test it with a few related functions. Some microservices, also known as high-level microservices, are mainly responsible for coordination with other microservices [124]. The organizational approach of microservices accelerates the development cycle, nourishes ownership, encourages innovation, improves scalability, and enhances the maintainability of software applications. Using this approach, software becomes a small independent service and interacts over unambiguous APIs. These services are preserved via self-contained small teams [125].

The agility and independent distributed nature of microservice deployment makes it a good solution for Fog-based IoT application development. Independent processes and interaction via message passing features has made microservices more convenient for IoT applications. In the Fog, there is a limitation of resources, so developing microservices in the Fog will minimize the growing complexity of the big system by dividing it into a set of small independent services. Microservice is taking modularity to a subsequent level by incorporating high cohesion and loose coupling of distributed systems.

1) CURRENT RESEARCH ASPECTS OF MICROSERVICE

Recently, microservice-based applications have started gaining popularity [124]. Fog-based microservices have not been investigated extensively; hence, it is an open research area. However, some research works have been done in this

TABLE 6. Summary of investigated fault tolerance literature in cloud.

Author & Year	Types of failure	Failure management	Mechanism employed	Infrastructure
Jiang and Hsu [114]	Hardware failure	Proactive	Two-level standby design	Cloud
Lati et al. [115]	Task failure	Reactive	Checkpointing and reallocation	Cloud
Wu et al. [116]	Host failure	Proactive	Migration	Cloud
Egwutuoha et al. [117]	Hardware /VM/ application failure	Proactive	Checkpointing and migration	Cloud/HPC
Sampaio and Barbosa [88]	Hardware, software and unstable behavior	Proactive	Prediction	Cloud
Gao et al. [120]	Task failure	Proactive and reactive	Crash detection, replication and migration	Cloud

TABLE 7. Simulation tools used for Fog simulation and their key features.

Name of the simulation tool	Proposed by	Key features	Usage example
CloudSim	Calheiros et al. [121]	A broad simulation toolkit that enables simulation and modeling of application provisioning in the cloud computing environments. The CloudSim toolkit supports system modeling of cloud system components such as virtual machines (VMs), data centers and resource allocation and provisioning policies as well as support system behavior modeling.	Aazam and Huh [54]; Dastjerdi et al. [4]
iFogSim	Gupta et al. [122]	Modelled IoT and Fog environments and measure the impact of resource management techniques in terms of network congestion, latency, cost and energy consumption. Developed on top of CloudSim	Baccarelli et al. [9]; Bittencourt et al. [124]; Markakis et al. [125]; Taneja and Davy [106]

emerging research area, with most of the efforts being related to IoT. Butzin *et al.* [126] investigated the use of microservices in IoT and claimed that the architectural goal of IoT and microservices are similar. However, they actually have different features in terms of various aspects. First of all, microservice has a self-containment feature where all dependencies and libraries are packed with the application in a single image. On the other hand, for IoT, all libraries are not generally wrapped with the application. However, both use similar types of virtualization and web protocols. Microservice also has a continuous integration and delivery feature while in IoT these are not available or only partly exist.

Vresk and Čavrak [127] proposed a microservice-based middleware for IoT to support device heterogeneity, various communication protocols, and services. They presented a data model and address model for microservice-based IoT. Brito *et al.* [128] proposed a service orchestration architecture for Fog using microservices. The authors defined the resource manager as a microservice. Khazaei *et al.* [129] proposed a generic programmable self-managing microservice-based platform for IoT. In the platform, microservices will exist in all layers in a cascading manner and an autonomic management system will scale the microservice. A similar type of IoT framework was proposed by Sun *et al.* [130]. In their architecture of nine components, all are microservices except for the core service. The work proved that microservices are far better than the monolithic approach in terms of scalability, flexibility, and platform independence. However, still, microservice-based IoT architecture suffers from various issues such as faults in the network, network delay, message serialization, cooperative transaction processing, and other distributed computing scenarios. Li *et al.* [131] proposed a cooperative-based model specifically for smart sensing devices; it is possible to enhance the performance of such a service by undertaking a micro-service based concept. Krivic *et al.* [132] proposed a management solution for Machine-to-Machine (M2M) device communication in an IoT system using collaborative microservice. They argued that microservices could act as the agent in an agent-based system since each microservice is responsible for performing a specific task and acts collaboratively to achieve the system goal.

Container-based virtualization is the best solution for deploying microservices since the container supports OS virtualization and packs all dependencies in a single image. A container is able to manage physical hardware resource needed by an application with its OS kernel utilities [133].

2) MICROSERVICES AND IOT APPLICATIONS

Many research works have suggested Fog-based processing for IoT applications in smart transportation

systems [96]–[98], Augmented and Virtual Reality [99], [134], [135], and healthcare [26], [136], [137]. Fog computing is also suitable for video streaming, smart homes, smart cities, and CDN. The common characteristics of these applications are time-sensitiveness, which make Fog computing a promising emerging computing paradigm. The main drawback of the Fog, however, is resource limitation and failure. Thus, using microservices for Fog-based IoT applications will minimize these drawbacks. Microservices are standalone, lightweight, and easily deliverable. To mitigate resource limitation, the microservice-based container is the best solution so far. In the same way, it will also minimize the cost of failure by deploying the application immediately. Many open research issues can be addressed by implementing Fog-based micro services; these include service management, scheduling, monitoring, fault tolerance, security, and privacy.

E. FOG BASED MOBILE COMPUTING

The number of smartphone and mobile device users in urban areas as well as in rural areas is increasing day by day. As a result, mobile users are now requesting high-volume content collaboratively. Providing service for all requested contents in an area where mobile users are densely populated is a really challenging task for service providers [138]. The high number of concurrent content requests will only make the situation worse. One of the best solutions to cope with this problem is to offload content near the users, so that the users could get better service. This content offloading process can be supported by mobile Fog computing. In mobile Fog computing, content will be offloaded to the Fog device, which is located closer to the users. However, content management in Fog nodes is a current research issue. Depending on the demand of the contents, offloading should be distributed on the Fog nodes. Constant monitoring and efficient cache management is crucial to deal with resource-limited Fog nodes. A few research works have been done on mobile Fog computing. This section chronologically discusses the research works that have been done in this particular research area in the past couple of years.

Hong *et al.* [94] proposed a high-level programming model that supports large-scale geospatially distributed time-sensitive applications. The proposed Mobile Fog programming model has two design goals. The initial goal is to provide a simplified application development for an enormous number of heterogenous devices, which are distributed in a wide area. The next goal is the dynamic scaling of resources based on resource demand. They developed an API for their programming model and evaluated it using two application models: vehicle tracking using a camera and traffic monitoring using a Mobility-driven distributed Complex Event Processing (MCEP) system. However, they did not focus on process placement or process migration.

Shi *et al.* [139] proposed a P2P inspired communication model between the mobile device cloud and mobile nodes to share resources and computation task among mobile devices. In their work, they utilized a Constrained Application Protocol (CoAP) for implementing microservices. This work introduced the M2M approach in Fog computing while the classical Fog is actually hierarchy based. Content offloading in the mobile Fog was investigated by Khan *et al.* [138]. They defined mobile Fog as co-located self-organizing mobile nodes, which offer distributed resources at the edge. The aim of this work was to collaborate nodes for content caching, which will maximize the availability of the content and minimize operational cost. The proposed coalition game helps find the best co-located candidates near the users for sharing storage and self-organizing.

Wang *et al.* [140] proposed a three-layer hierarchy framework using a Fog structure to bridge the communication between WSNs and the cloud. They designed a routing algorithm for bridging communication by considering the number of hops and energy consumption. They defined the Fog node as a sink, which will transfer data from sensor to the cloud. The proposed framework consists of routing layer, Fog layer, and sink layer. In the Fog layer, a sink acts as the Fog nodes as well to minimize transmission delay. However, there is a lack of security and privacy concern being addressed in mobile Fog computing. Roman *et al.* [141] addressed security and privacy for all edge-level computing. This included a usual thread in a network system mobile Fog that requires extra measures of authentication, trust, access control, protocol, and network security.

IX. OPEN ISSUES AND FUTURE RESEARCH DIRECTIONS

A. INFRASTRUCTURE-RELATED ISSUES

The Fog is an evolving technology, expanding in such a way that it needs to reach market adoption to support all kinds of time-sensitive applications. The Fog has become an enactment of research efforts by various academies and industries. One of the key initiatives is the Open Fog Consortium (OpenFog), which was founded by the ARM, Cisco, Dell, Intel, Microsoft and Princeton University in November 2015 [31]. Foxconn, General Electric, Hitachi, Sakura Internet, ShanghaiTech University, and ZTE are the contributing members of this nonprofit consortium. They are accelerating digital innovation with the blending of 5G wireless technology, IoT, and embedded AI by providing open interoperable architecture. However, many open challenges exist for this sprout-level computing paradigm. In this section, we discuss the research challenges and address the future directions for Fog computing research. Figure 10 shows some important research issues in Fog computing.

1) DEPLOYMENT ISSUES

From the deployment viewpoint, OpenFog is defined as an N-tier environment. However, the excessive increase in number of levels in the Fog layer might cause latency problems in the newly emerging Fog computing paradigm. Therefore, the number of tiers based on the use case must be determined. Deployment decisions will be undertaken based on requirements such as type and amount of task that will be done

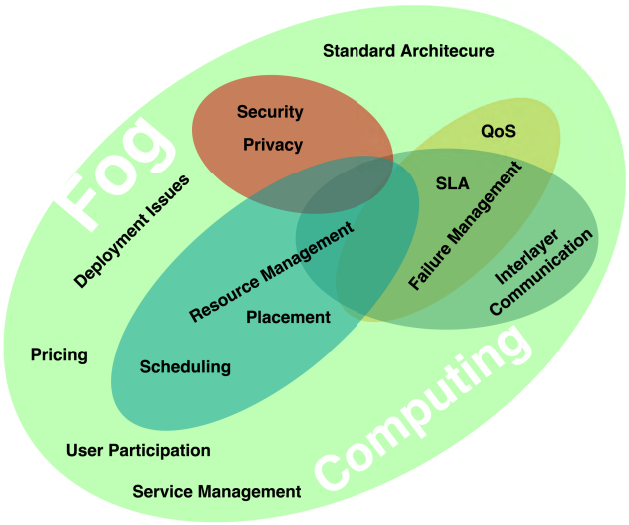

FIGURE 10. Fog computing research issues.

by each tier, total number of sensors, Fog device capability, in between the latency and reliability of Fog devices. Still, it is necessary to investigate how these requirements will be fulfilled. Application and resource scaling is also an important issue during deployment. Based on the requirement of the application and resource, scaling and shrinking without interrupting current services could be undertaken. In this regard, placement might also affect Fog deployment.

2) STANDARD ARCHITECTURE FOR FOG COMPUTING

Up until now, there has been no defined standard architecture for Fog computing. The OpenFog consortium released two versions of the Fog architecture in February 2016 [61] and February 2017 [142]. Their first draft was an initial overview of the Fog architecture. In their second draft, the Fog architecture was discussed in more detail. In their proposed architecture, they considered many key aspects of Fog architecture including performance, manageability, security, data analytics, and control. However, further research needs to be undertaken to explore and gain deeper insight into each layer with proper validation.

3) INTEROPERABILITY AND FEDERATION OF FOG

Because of the Fog, users are able to process their request near them, which will minimize latency. However, what will happen if an increasing number of multiple latency-aware applications requests are sent in one shot to the Fog device and the Fog device is unable to handle that many requests? Will it be passed to the cloud for processing? If it is passed to the cloud, then latency requirements will not be satisfied. Thus, interoperability and federation among Fog clusters and Fog servers are necessary. Hence, if a Fog device is fully utilized, it will send requests to peer Fog devices or Fog servers for processing instead of sending them to the cloud.

B. PLATFORM-RELATED ISSUES

1) RESOURCE MANAGEMENT

Resources are most dynamic and heterogeneous in a Fog environment because of the diversity of devices and their available resources. All devices known as Fog devices are responsible for performing the computation of their own application. For example, a computer that relies on office staff to perform some ordinary email and documentation works might be a part of the Fog and might also act as a Fog device. In such a case, the amount of resources available for Fog computation is dynamic but predictable via the analysis of the long-term activity of its resources. This prediction is necessary because once the Fog task execution starts, and over a period of time, the status of the resources might change due to the request by the application for which the device is responsible for. If we compared this to the cloud, it is possible to know how much resources are currently available and whether or not they are exclusively used for cloud-based application requests. However, the Fog aims to use idle resources available on any Fog device with Fog computation always taking second priority. Hence, resource allocation and scheduling in the Fog is more challenging than traditional resource allocation and scheduling in the cloud.

2) FAILURE MANAGEMENT

Fog device failure probability is always high because the devices are distributed and the management of Fog devices is not central. Hence, the devices could fail for many reasons; this could be due to hardware failure, software failure, or because of user activity. Besides these problems, some other reasons include connectivity, mobility, and power source, which also play a big role. Most of the devices in a Fog environment might be connected via wireless connections; it is obvious that wireless connections are not always reliable. The majority of devices that are connected via wireless are mobile, so these devices could change location to different clusters frequently. One other characteristic of these devices is that they are battery powered and might fail anytime. Hence, dealing with the complex nature of failure is very difficult. Also, it is necessary to ensure SLA by defining QoS parameters. So, the question is: What are the SLAs and how should they be defined? Also: What QoS parameters must be considered, so that the consumer and providers can retain a win-win situation?

3) COMMUNICATION BETWEEN DIFFERENT LAYERS

The Fog should ensure uninterrupted connection with the devices to ensure application requirements of time-sensitive applications are met. If the application were to control an autonomous car or drone and if it were responsible for emergency surveillance, then failure in connectivity might cause serious harm. Even if connectivity to the cloud fails, the Fog still needs to ensure continuous connectivity. Thus, cross-layer connectivity among IoT devices, Fog, and cloud are of the utmost importance. The connection type and

protocols used by IoT devices and Fog devices might be different. Therefore, how these issues will be handled is an important research issue.

4) USER PARTICIPATION MANAGEMENT

Efficient Fog service management depends on the participation of users in Fog computation. However, how can user participation be managed? How do we deploy minimum resources in the case where no one wants to participate? We need to address these problems clearly with a feasible solution. One of the methods to increase user participation is through incentive and reward-based policies. With such policies, any user that participates in Fog computation will benefit. Even a user, who participates to complete his own request, will be rewarded by getting some discount based on his participation. However, this area needs to be addressed because the overall success of Fog computation depends on the participation of Fog devices, which are owned by various people and organizations.

5) SECURITY AND PRIVACY

Fog devices are managed by different operators based on their location and ownership. Nobody would want to contribute to Fog computation if device control were compromised. Thus, how the security of a participant device will be maintained if the device were to take part in Fog computation is a big question. Another key security issue for this scenario is participant user data security. A participant device might have critical information. How will safety be guaranteed in such a case? On the other hand, critical data might be processed in a device, which is owned by a black hat hacker. How will safety and privacy be ensured then? Security issues also exist during cross-layer communication. Similar to the distributed nature of the Fog, security management should also be distributed, which will not be dependent on any central component.

C. APPLICATION-RELATED ISSUES

1) APPLICATION SERVICE MANAGEMENT

Billions of IoT devices will be handled by the Fog paradigm, which will handle time-sensitive and time-insensitive applications. The degree of service, availability, and quality is most diverse in the Fog. Hence, service management is a typical issue for the entire Fog realm. Services should be microservice-based, so that agility and management issues can be handled properly. Further research is necessary to explore the possibility of Fog-based solutions.

2) APPLICATION MODELING

Modeling Fog applications is complex because the application should collect data from different IoT devices, which use different protocols and sets of codes. Thus, it is challenging to model generic applications, which can be deployed with minimal effort. To solve this issue, a standard form of communication protocol is necessary, so that the modeled application can communicate and work with different types of IoT devices.

X. CONCLUSION

The Fog computing paradigm is currently in its infancy, so an extensive investigation is required for this emerging technology. In this survey, we presented and discussed the overview, architecture, state-of-the-art and other similar technologies in Fog computing. Based on the literature, we derived a taxonomy for Fog computing by analyzing the requirement of Fog infrastructure, platform, and applications. We also covered resource allocation and scheduling, fault tolerance, simulation tools, and microservices in Fog computing. Finally, we presented some challenging and open research issues. We strongly believe that this comprehensive survey will bring to light IoT application execution for a Fog computing environment as well as point towards the direction for current and future research in this rapidly growing research area. In this way, this computing paradigm, which is still immature, will be propelled towards achieving market adoption in the near future.

ACKNOWLEDGEMENT

The authors would like to sincerely thank Professor Rajkumar Buyya for his insightful comments and discussion on improving the overall quality of the paper.

REFERENCES

[1] M. D. Assunção, R. N. Calheiros, S. Bianchi, M. A. S. Netto, and R. Buyya, "Big data computing and clouds: Trends and future directions," *J. Parallel Distrib. Comput.*, vols. 79–80, pp. 3–15, May 2015.
[2] J. Chen *et al.*, "Big data challenge: A data management perspective," *Frontiers Comput. Sci.*, vol. 7, no. 2, pp. 157–164, 2013.
[3] F. Alhaddadin, W. Liu, and J. A. Gutiérrez, "A user profile-aware policy-based management framework for greening the cloud," in *Proc. IEEE 4th Int. Conf. Big Data Cloud Comput. (BdCloud)*, Dec. 2014, pp. 682–687.
[4] A. V. Dastjerdi, H. Gupta, R. N. Calheiros, S. K. Ghosh, and R. Buyya, "Fog computing: Principles, architectures, and applications," in *Internet of Things: Principle & Paradigms*. San Mateo, CA, USA: Morgan Kaufmann, 2016.
[5] R. Mahmud, R. Kotagiri, and R. Buyya, "Fog computing: A taxonomy, survey and future directions," in *Internet of Everything*. Singapore: Springer, 2018, pp. 103–130.
[6] L. Gao, T. H. Luan, S. Yu, W. Zhou, and B. Liu, "FogRoute: DTN-based data dissemination model in fog computing," *IEEE Internet Things J.*, vol. 4, no. 1, pp. 225–235, Feb. 2017.
[7] S. Yi, C. Li, and Q. Li, "A survey of fog computing: Concepts, applications and issues," in *Proc. Workshop Mobile Big Data*, 2015, pp. 37–42.
[8] P. Hu, S. Dhelim, H. Ning, and T. Qiu, "Survey on fog computing: Architecture, key technologies, applications and open issues," *J. Netw. Comput. Appl.*, vol. 98, pp. 27–42, Nov. 2017.
[9] E. Baccarelli, P. G. V. Naranjo, M. Scarpiniti, M. Shojafar, and J. H. Abawajy, "Fog of everything: Energy-efficient networked computing architectures, research challenges, and a case study," *IEEE Access*, vol. 5, pp. 9882–9910, 2017.
[10] P. Varshney and Y. Simmhan, "Demystifying fog computing: Characterizing architectures, applications and abstractions," in *Proc. IEEE 1st Int. Conf. Fog Edge Comput. (ICFEC)*, Feb. 2017, pp. 115–124.
[11] C. Perera, Y. Qin, J. C. Estrella, S. Reiff-Marganiec, and A. V. Vasilakos, "Fog computing for sustainable smart cities: A survey," *ACM Comput. Surv.*, vol. 50, no. 3, p. 32, Oct. 2017.
[12] C. Mouradian, D. Naboulsi, S. Yangui, R. H. Glitho, M. J. Morrow, and P. A. Polakos, "A comprehensive survey on fog computing: State-of-the-art and research challenges," *IEEE Commun. Surveys Tuts.*, vol. 20, no. 1, pp. 416–464, 1st Quart., 2018.

[13] F. Bonomi, R. Milito, J. Zhu, and S. Addepalli, "Fog computing and its role in the Internet of Things," in *Proc. 1st Ed. MCC Workshop Mobile Cloud Comput.*, 2012, pp. 13–16.
[14] X. Xie, H.-J. Zeng, and W.-Y. Ma, "Enabling personalization services on the edge," in *Proc. 10th ACM Int. Conf. Multimedia*, 2002, pp. 263–266.
[15] P. P. Gelsinger, "Microprocessors for the new millennium: Challenges, opportunities, and new frontiers," in *Proc. IEEE Int. Solid-State Circuits Conf.*, Feb. 2001, pp. 22–25.
[16] S. Ibrahim, H. Jin, B. Cheng, H. Cao, S. Wu, and L. Qi, "CLOUDLET: Towards mapreduce implementation on virtual machines," in *Proc. 18th ACM Int. Symp. High Perform. Distrib. Comput.*, 2009, pp. 65–66.
[17] N. M. Gonzalez *et al.*, "Fog computing: Data analytics and cloud distributed processing on the network edges," in *Proc. 35th Int. Conf. Chilean Comput. Sci. Soc. (SCCC)*, Oct. 2016, pp. 1–9.
[18] J. Li, T. Zhang, J. Jin, Y. Yang, D. Yuan, and L. Gao, "Latency estimation for fog-based Internet of Things," in *Proc. 27th Int. Telecommun. Netw. Appl. Conf. (ITNAC)*, Nov. 2017, pp. 1–6.
[19] "Fog computing and the Internet of Things: Extend the cloud to where the things are," Cisco, San Jose, CA, USA, White Paper, 2015. [Online]. Available: https://www.cisco.com/c/dam/en_us/solutions/trends/iot/../computing-overview.pdf
[20] L. M. Vaquero and L. Rodero-Merino, "Finding your way in the fog: Towards a comprehensive definition of fog computing," *ACM SIGCOMM Comput. Commun. Rev.*, vol. 44, no. 5, pp. 27–32, Oct. 2014.
[21] (Sep. 2016). *What is Fog Computing?* [Online]. Available: https://www.ibm.com/blogs/cloud-computing/2014/08/fog-computing/
[22] Cisco Systems. (2016). *Fog Computing and the Internet of Things: Extend the Cloud to Where the Things Are*. [Online]. Available: http://www.cisco.com/c/dam/en_us/solutions/trends/iot/docs/computingoverview.pdf
[23] W. Shi, J. Cao, Q. Zhang, Y. Li, and L. Xu, "Edge computing: Vision and challenges," *IEEE Internet Things J.*, vol. 3, no. 5, pp. 637–646, Oct. 2016.
[24] (Aug. 2017). *Top Trends in the Gartner Hype Cycle for Emerging Technologies, 2017*. [Online]. Available: http://www.gartner.com/smarterwithgartner/top-trends-in-the-gartner-hype-cycle-for-emerging-technologies-2017/
[25] R. Buyya, C. S. Yeo, S. Venugopal, J. Broberg, and I. Brandic, "Cloud computing and emerging IT platforms: Vision, hype, and reality for delivering computing as the 5th utility," *Future Generat. Comput. Syst.*, vol. 25, no. 6, pp. 599–616, 2009.
[26] R. Mahmud, F. L. Koch, and R. Buyya, "Cloud-fog interoperability in IoT-enabled healthcare solutions," in *Proc. 19th Int. Conf. Distrib. Comput. Netw. (ICDCN)*. New York, NY, USA: ACM, 2018, pp. 32:1–32:10.
[27] T. H. Luan, L. Gao, Z. Li, Y. Xiang, G. Wei, and L. Sun. (Mar. 2016). "Fog computing: Focusing on mobile users at the edge." pp. 1–11. [Online]. Available: https://arxiv.org/abs/1502.01815
[28] Y. Wang, T. Uehara, and R. Sasaki, "Fog computing: Issues and challenges in security and forensics," in *Proc. IEEE 39th Annu. Comput. Softw. Appl. Conf. (COMPSAC)*, vol. 3, Jul. 2015, pp. 53–59.
[29] M. H. Syed, E. B. Fernandez, and M. Ilyas, "A pattern for fog computing," in *Proc. 10th Travelling Conf. Pattern Lang. Programs*, 2016, p. 13.
[30] S. Yi, Z. Hao, Z. Qin, and Q. Li, "Fog computing: Platform and applications," in *Proc. 3rd IEEE Workshop Hot Topics Web Syst. Technol. (HotWeb)*, Nov. 2015, pp. 73–78.
[31] M. Chiang and T. Zhang, "Fog and IoT: An overview of research opportunities," *IEEE Internet Things J.*, vol. 3, no. 6, pp. 854–864, Dec. 2016.
[32] Z. Sanaei, S. Abolfazli, A. Gani, and R. Buyya, "Heterogeneity in mobile cloud computing: Taxonomy and open challenges," *IEEE Commun. Surveys Tuts.*, vol. 16, no. 1, pp. 369–392, Feb. 2014.
[33] P. Bahl, R. Y. Han, L. E. Li, and M. Satyanarayanan, "Advancing the state of mobile cloud computing," in *Proc. 3rd ACM Workshop Mobile Cloud Comput. Services*, 2012, pp. 21–28.
[34] M. Satyanarayanan, G. Lewis, E. Morris, S. Simanta, J. Boleng, and K. Ha, "The role of cloudlets in hostile environments," *IEEE Pervas. Comput.*, vol. 12, no. 4, pp. 40–49, Oct. 2013.
[35] M. T. Beck, M. Werner, S. Feld, and T. Schimper, "Mobile edge computing: A taxonomy," in *Proc. 6th Int. Conf. Adv. Future Internet*, 2014, pp. 48–54.
[36] G. I. Klas. (2015). *Fog Computing and Mobile Edge Cloud Gain Momentum Open Fog Consortium ETSI MEC and Cloudlets*. [Online]. Available: http://yucianga.info/?p=938
[37] M. Satyanarayanan, P. Bahl, R. Caceres, and N. Davies, "The case for VM-based cloudlets in mobile computing," *IEEE Pervasive Comput.*, vol. 8, no. 4, pp. 14–23, Oct./Dec. 2009.
[38] Y. Wang, "Cloud-dew architecture," *Int. J. Cloud Comput.*, vol. 4, no. 3, pp. 199–210, 2015.
[39] K. Skala, D. Davidović, E. Afgan, I. Sović, and Z. Šojat, "Scalable distributed computing hierarchy: Cloud, fog and dew computing," *Open J. Cloud Comput.*, vol. 2, no. 1, pp. 16–24, Mar. 2015.
[40] Y. Wang and Y. Pan, "Cloud-dew architecture: Realizing the potential of distributed database systems in unreliable networks," in *Proc. Int. Conf. Parallel Distrib. Process. Technol. Appl. (PDPTA)*, 2015, p. 85.
[41] T. Mane. (2017). *Fog-Dew Architecture for Better Consistency*. [Online]. Available: https://eye3i.wordpress.com/2016/07/02/adressing-inconsistency-in-dew-computing-using-fog-computing/
[42] (2018). *Akamai Mobile Cloud Computing*. [Online]. Available: https://www.akamai.com/us/en/resources/mobile-cloud-computing.jsp
[43] (2018). *EC-IoT Solution*. [Online]. Available: http://e.huawei.com/en/solutions/technical/sdn/agile-iot
[44] (Mar. 2016). *Cisco IOx Local Manager Pages and Options*. http://www.cisco.com/c/en/us/td/docs/routers/access/800/software/guides/iox/lm/reference-guide/1-0/iox_local_manager_ref_guide/ui_reference.html
[45] (2018). *Hyrax Crowd-Sourcing Mobile Devices to Develop Edge Clouds*. [Online]. Available: http://hyrax.dcc.fc.up.pt/
[46] (2018). *Open-RAN Mobile Edge Computing Platform*. [Online]. Available: https://www.sdxcentral.com/products/saguna-open-ran-mobile-edge-computing-platform/
[47] H. Dubey, J. Yang, N. Constant, A. M. Amiri, Q. Yang, and K. Makodiya, "Fog data: Enhancing telehealth big data through fog computing," in *Proc. ASE BigData SocialInform.*, 2015, p. 14.
[48] M. Ahmad, M. Bilal, S. Hussain, B. Ho, T. Cheong, and S. Lee, "Health fog: A novel framework for health and wellness applications," *J. Supercomput.*, vol. 72, no. 10, pp. 3677–3695, 2016.
[49] B. Tang *et al.*, "Incorporating intelligence in fog computing for big data analysis in smart cities," *IEEE Trans. Ind. Informat.*, vol. 13, no. 5, pp. 2140–2150, Oct. 2017.
[50] B. Tang, Z. Chen, G. Hefferman, T. Wei, H. He, and Q. Yang, "A hierarchical distributed fog computing architecture for big data analysis in smart cities," in *Proc. ASE BigData SocialInform.*, 2015, p. 28.
[51] W. Zhang, Z. Zhang, and H.-C. Chao, "Cooperative fog computing for dealing with big data in the Internet of vehicles: Architecture and hierarchical resource management," *IEEE Commun. Mag.*, vol. 55, no. 12, pp. 60–67, Dec. 2017.
[52] B. Yin, W. Shen, Y. Cheng, L. X. Cai, and Q. Li, "Distributed resource sharing in fog-assisted big data streaming," in *Proc. IEEE Int. Conf. Commun.*, May 2017, pp. 1–6.
[53] R. Pecori, "A virtual learning architecture enhanced by fog computing and big data streams," *Future Internet*, vol. 10, no. 1, pp. 1–30, 2018.
[54] M. Aazam and E.-N. Huh, "Fog computing micro datacenter based dynamic resource estimation and pricing model for IoT," in *Proc. 29th IEEE Int. Conf. Adv. Inf. Netw. Appl. (AINA)*, Mar. 2015, pp. 687–694.
[55] H. R. Arkian, A. Diyanat, and A. Pourkhalili, "MIST: Fog-based data analytics scheme with cost-efficient resource provisioning for IoT crowdsensing applications," *J. Netw. Comput. Appl.*, vol. 82, pp. 152–165, Mar. 2017.
[56] M. A. Nadeem and M. A. Saeed, "Fog computing: An emerging paradigm," in *Proc. 6th Int. Conf. Innov. Comput. Technol. (INTECH)*, Aug. 2016, pp. 83–86.
[57] M. Taneja and A. Davy, "Resource aware placement of data analytics platform in fog computing," *Procedia Comput. Sci.*, vol. 97, pp. 153–156, Jan. 2016.
[58] S. Sarkar and S. Misra, "Theoretical modelling of fog computing: A green computing paradigm to support IoT applications," *IET Netw.*, vol. 5, no. 2, pp. 23–29, 2016.
[59] N. K. Giang, M. Blackstock, R. Lea, and V. C. M. Leung, "Developing IoT applications in the fog: A distributed dataflow approach," in *Proc. 5th Int. Conf. Internet Things (IOT)*, Oct. 2015, pp. 155–162.
[60] F. Hosseinpour, J. Plosila, and H. Tenhunen, "An approach for smart management of big data in the fog computing context," in *Proc. IEEE Int. Conf. Cloud Comput. Technol. Sci. (CloudCom)*, Dec. 2016, pp. 468–471.
[61] OpenFog Consortium Architecture Working Group, "OpenFog architecture overview," OpenFog Consortium, Tokyo, Japan, White Paper OPFWP001.0216, Feb. 2016.
[62] K. Intharawijitr, K. Iida, and H. Koga, "Analysis of fog model considering computing and communication latency in 5G cellular networks," in *Proc. IEEE Int. Conf. Pervasive Comput. Commun. Workshops (PerCom Workshops)*, Mar. 2016, pp. 1–4.

[63] Y. Sun and N. Zhang, "A resource-sharing model based on a repeated game in fog computing," *Saudi J. Biol. Sci.*, vol. 24, no. 3, pp. 687–694, 2017.

[64] A. Munir, P. Kansakar, and S. U. Khan, "IFCIoT: Integrated fog cloud IoT: A novel architectural paradigm for the future Internet of Things," *IEEE Consum. Electron. Mag.*, vol. 6, no. 3, pp. 74–82, Jul. 2017.

[65] (Dec. 2016). *Cisco Fog Data Services*. [Online]. Available: http://www.cisco.com/c/en/us/products/cloud-systems-management/fog-data-services/index.html

[66] (Jan. 2017). *Cisco Fog Director*. [Online]. Available: http://www.cisco.com/c/en/us/products/cloud-systems-management/fog-director/index.html

[67] G. Albeanu and F. Popentiu-Vladicescu, "A reliable e-learning architecture based on fog-computing and smart devices," in *Proc. Int. Sci. Conf. eLearn. Softw. Edu.*, vol. 4. 2014, p. 9.

[68] Metro Network Traffic Growth, "An architecture impact study," Alcatel-Lucent Bell Labs, Murray Hill, NJ, USA, White Paper NP2013113265EN, 2013.

[69] S. Grehl *et al.*, "Research perspective-mobile robots in underground mining," in *Proc. AusIMM Bull.*, Feb. 2017, p. 44.

[70] V. Turner. (2016). *IDC Link: IBM and Cisco Bridge the IoT Edge and Enterprise*. pp. 1–2. [Online]. Available: https://www-01.ibm.com/common/ssi/cgi-bin/ssialias?htmlfid=WWL12359USEN

[71] J. Biswas, F. Naumann, and Q. Qiu, "Assessing the completeness of sensor data," in *Proc. Int. Conf. Database Syst. Adv. Appl.* Berlin, Germany: Springer, 2006, pp. 717–732.

[72] M. Kocakulak and I. Butun, "An overview of Wireless Sensor Networks towards Internet of Things," in *Proc. 7th IEEE Annu. Comput. Commun. Workshop Conf. (CCWC)*, Jan. 2017, pp. 1–6.

[73] K. Aziz, S. Tarapiah, S. H. Ismail, and S. Atalla, "Smart real-time healthcare monitoring and tracking system using GSM/GPS technologies," in *Proc. 3rd MEC Int. Conf. Big Data Smart City (ICBDSC)*, Mar. 2016, pp. 1–7.

[74] A. H. Butt, A. Moschetti, L. Fiorini, P. Dario, and F. Cavallo, "Wearable sensors for gesture analysis in smart healthcare applications," in *Human Monitoring, Smart Health and Assisted Living: Techniques and Technologies*. Rijeka, Croatia: InTech, 2017, p. 79.

[75] N. Constant, D. Borthakur, M. Abtahi, H. Dubey, and K. Mankodiya. (2017). "Fog-assisted wIoT: A smart fog gateway for end-to-end analytics in wearable Internet of Things." [Online]. Available: https://arxiv.org/abs/1701.08680

[76] B. Kang, D. Kim, and H. Choo, "Internet of everything: A large-scale autonomic IoT gateway," *IEEE Trans. Multi-Scale Comput. Syst.*, vol. 3, no. 3, pp. 206–214, Jul./Sep. 2017.

[77] S. Guoqiang, C. Yanming, Z. Chao, and Z. Yanxu, "Design and implementation of a smart IoT gateway," in *Proc. IEEE Int. Conf. Green Comput. Commun. IEEE Internet Things IEEE Cyber, Phys. Social Comput.*, Aug. 2013, pp. 720–723.

[78] Z. Amin, H. Singh, and N. Sethi, "Review on fault tolerance techniques in cloud computing," *Int. J. Comput. Appl.*, vol. 116, no. 18, pp. 1–17, 2015.

[79] D. Poola, S. K. Garg, R. Buyya, Y. Yang, and K. Ramamohanarao, "Robust scheduling of scientific workflows with deadline and budget constraints in clouds," in *Proc. 28th IEEE Int. Conf. Adv. Inf. Netw. Appl. (AINA)*, May 2014, pp. 858–865.

[80] S. Paul, R. Jain, M. Samaka, and J. Pan, "Application delivery in multi-cloud environments using software defined networking," *Comput. Netw.*, vol. 68, pp. 166–186, Aug. 2014.

[81] R. Jhawar, V. Piuri, and M. Santambrogio, "A comprehensive conceptual system-level approach to fault tolerance in cloud computing," in *Proc. IEEE Int. Syst. Conf. (SysCon)*, Mar. 2012, pp. 1–5.

[82] W. Zhao, P. Melliar-Smith, and L. E. Moser, "Fault tolerance middleware for cloud computing," in *Proc. 3rd IEEE Int. Conf. Cloud Comput. (CLOUD)*, Jul. 2010, pp. 67–74.

[83] R. Jhawar, V. Piuri, and M. Santambrogio, "Fault tolerance management in cloud computing: A system-level perspective," *IEEE Syst. J.*, vol. 7, no. 2, pp. 288–297, Jun. 2013.

[84] C.-T. Yang, J.-C. Liu, C.-H. Hsu, and W.-L. Chou, "On improvement of cloud virtual machine availability with virtualization fault tolerance mechanism," *J. Supercomput.*, vol. 69, no. 3, pp. 1103–1122, 2014.

[85] J. Liu, J. Zhou, and R. Buyya, "Software rejuvenation based fault tolerance scheme for cloud applications," in *Proc. 8th IEEE Int. Conf. Cloud Comput. (CLOUD)*, Jun./Jul. 2015, pp. 1115–1118.

[86] B. Mohammed, M. Kiran, K. M. Maiyama, M. M. Kamala, and I.-U. Awan, "Failover strategy for fault tolerance in cloud computing environment," *Softw., Pract. Exper.*, vol. 47, no. 9, pp. 1243–1274, 2017.

[87] J. Liu, S. Wang, A. Zhou, S. Kumar, F. Yang, and R. Buyya, "Using proactive fault-tolerance approach to enhance cloud service reliability," *IEEE Trans. Cloud Comput.*, to be published.

[88] A. M. Sampaio and J. G. Barbosa, "A comparative cost study of fault-tolerant techniques for availability on the cloud," in *Proc. Int. Symp. Ambient Intell.* Springer, 2017, pp. 263–268.

[89] H.-W. Kim, G. Yi, J. H. Park, and Y.-S. Jeong, "Adaptive resource management using many-core processing for fault tolerance based on cyber–physical cloud systems," *Future Generat. Comput. Syst.*, to be published.

[90] H. Madsen, B. Burtschy, G. Albeanu, and F. Popentiu-Vladicescu, "Reliability in the utility computing era: Towards reliable fog computing," in *Proc. 20th Int. Conf. Syst., Signals Image Process. (IWSSIP)*, Jul. 2013, pp. 43–46.

[91] E. Aghasian, S. Garg, L. Gao, S. Yu, and J. Montgomery, "Scoring users' privacy disclosure across multiple online social networks," *IEEE Access*, vol. 5, pp. 13118–13130, 2017.

[92] (Mar. 2018). *Here's Everything You Need to Know About the Cambridge Analytica Scandal*. [Online]. Available: https://www.cnbc.com/2018/03/21/facebook-cambridge-analytica-scandal-everything-you-need-to-know.html

[93] F. Li, M. Vöegler, M. Claeßens, and S. Dustdar, "Efficient and scalable IoT service delivery on cloud," in *Proc. IEEE 6th Int. Conf. Cloud Comput. (CLOUD)*, Jun./Jul. 2013, pp. 740–747.

[94] K. Hong, D. Lillethun, U. Ramachandran, B. Ottenwälder, and B. Koldehofe, "Mobile fog: A programming model for large-scale applications on the Internet of Things," in *Proc. 2nd ACM SIGCOMM Workshop Mobile Cloud Comput.*, 2013, pp. 15–20.

[95] A. V. Dastjerdi and R. Buyya, "Fog computing: Helping the Internet of Things realize its potential," *Computer*, vol. 49, no. 8, pp. 112–116, Aug. 2016.

[96] N. B. Truong, G. M. Lee, and Y. Ghamri-Doudane, "Software defined networking-based vehicular adhoc network with fog computing," in *Proc. IFIP/IEEE Int. Symp. Integr. Netw. Manage. (IM)*, May 2015, pp. 1202–1207.

[97] N. K. Giang, V. C. M. Leung, and R. Lea, "On developing smart transportation applications in fog computing paradigm," in *Proc. 6th ACM Symp. Develop. Anal. Intell. Veh. Netw. Appl.*, 2016, pp. 91–98.

[98] X. Hou, Y. Li, M. Chen, D. Wu, D. Jin, and S. Chen, "Vehicular fog computing: A viewpoint of vehicles as the infrastructures," *IEEE Trans. Veh. Technol.*, vol. 65, no. 6, pp. 3860–3873, Jun. 2016.

[99] J. K. Zao *et al.*, "Augmented brain computer interaction based on fog computing and linked data," in *Proc. Int. Conf. Intell. Environ. (IE)*, Jun./Jul. 2014, pp. 374–377.

[100] A. M. Rahmani *et al.*, "Exploiting smart e-health gateways at the edge of healthcare Internet-of-Things: A fog computing approach," *Future Gener. Comput. Syst.*, vol. 78, pp. 641–658, Jan. 2018.

[101] A. Giordano, G. Spezzano, and A. Vinci, "Smart agents and fog computing for smart city applications," in *Proc. Int. Conf. Smart Cities*. London, U.K.: Springer, 2016, pp. 137–146.

[102] A. A. Alsaffar, H. P. Pham, C.-S. Hong, E.-N. Huh, and M. Aazam, "An architecture of iot service delegation and resource allocation based on collaboration between fog and cloud computing," *Mobile Inf. Syst.*, vol. 2016, Aug. 2016, Art. no. 6123234.

[103] R. Deng, R. Lu, C. Lai, T. H. Luan, and H. Liang, "Optimal workload allocation in fog-cloud computing toward balanced delay and power consumption," *IEEE Internet Things J.*, vol. 3, no. 6, pp. 1171–1181, Dec. 2016.

[104] A. Brogi, S. Forti, and A. Ibrahim, "How to best deploy your fog applications, probably," in *Proc. Int. Conf. Edge Fog Comput.*, May 2017, pp. 105–114.

[105] M. Taneja and A. Davy, "Resource aware placement of IoT application modules in fog-cloud computing paradigm," in *Proc. IFIP/IEEE Symp. Integr. Netw. Service Manage. (IM)*, May 2017, pp. 1222–1228.

[106] M. Aazam, M. St-Hilaire, C.-H. Lung, I. Lambadaris, and E.-N. Huh, "IoT resource estimation challenges and modeling in fog," in *Fog Computing in the Internet of Things*. Springer, 2018, pp. 17–31.

[107] L. Ni, J. Zhang, C. Jiang, C. Yan, and K. Yu, "Resource allocation strategy in fog computing based on priced timed petri nets," *IEEE Internet Things J.*, vol. 4, no. 5, pp. 1216–1228, 2017.

[108] Z. Pooranian, M. Shojafar, P. G. V. Naranjo, L. Chiaraviglio, and M. Conti, "A novel distributed fog-based networked architecture to preserve energy in fog data centers," in *Proc. 2017 IEEE 14th Int. Conf. Mobile Ad Hoc Sensor Syst. (MASS)*, Orlando, FL, USA, 2017, pp. 22–25.
[109] B. Urgaonkar, G. Pacifici, P. Shenoy, M. Spreitzer, and A. Tantawi, "Analytic modeling of multitier Internet applications," *ACM Trans. Web*, vol. 1, no. 1, p. 2, 2007.
[110] Y. Sharma, B. Javadi, W. Si, and D. Sun, "Reliability and energy efficiency in cloud computing systems: Survey and taxonomy," *J. Netw. Comput. Appl.*, vol. 74, pp. 66–85, Oct. 2016.
[111] Y. Liu, J. E. Fieldsend, and G. Min, "A framework of fog computing: Architecture, challenges, and optimization," *IEEE Access*, vol. 5, pp. 25445–25454, 2017.
[112] F.-C. Jiang and C.-H. Hsu, "Fault-tolerant system design on cloud logistics by greener standbys deployment with Petri net model," *Neurocomputing*, vol. 256, pp. 90–100, Sep. 2017.
[113] M. S. A. Latiff *et al.*, "A checkpointed league championship algorithm-based cloud scheduling scheme with secure fault tolerance responsiveness," *Appl. Soft Comput.*, vol. 61, pp. 670–680, Dec. 2017.
[114] Y. Wu, H. Song, Y. Xiong, Z. Zheng, Y. Zhang, and G. Huang, "Model defined fault tolerance in cloud," in *Proc. 6th Asia–Pacific Symp. Internetware Internetware*, 2014, pp. 116–119.
[115] I. P. Egwutuoha, S. Chen, D. Levy, B. Selic, and R. Calvo, "Energy efficient fault tolerance for high performance computing (HPC) in the cloud," in *Proc. IEEE 6th Int. Conf. Cloud Comput. (CLOUD)*, Jun./Jul. 2013, pp. 762–769.
[116] S. Fu, "Failure-aware resource management for high-availability computing clusters with distributed virtual machines," *J. Parallel Distrib. Comput.*, vol. 70, no. 4, pp. 384–393, 2010.
[117] A. M. Sampaio and J. G. Barbosa, "Towards high-available and energy-efficient virtual computing environments in the cloud," *Future Gener. Comput. Syst.*, vol. 40, pp. 30–43, Nov. 2014.
[118] Y. Gao, S. K. Gupta, Y. Wang, and M. Pedram, "An energy-aware fault tolerant scheduling framework for soft error resilient cloud computing systems," in *Proc. Design, Automat. Test Eur. Conf. Exhib. (DATE)*, Mar. 2014, pp. 1–6.
[119] R. Calheiros, R. Ranjan, A. Beloglazov, C. A. F. De Rose, and R. Buyya, "CloudSim: A toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms," *Softw., Pract. Exper.*, vol. 41, no. 1, pp. 23–50, 2011.
[120] H. Gupta, A. V. Dastjerdi, S. K. Ghosh, and R. Buyya, "iFogSim: A toolkit for modeling and simulation of resource management techniques in Internet of Things, edge and fog computing environments," *Softw., Pract. Exper.*, vol. 47, no. 9, pp. 1275–1296, 2017.
[121] L. Gao, T. H. Luan, B. Liu, W. Zhou, and S. Yu, "Fog computing and its applications in 5G," in *Proc. 5G Mobile Commun.* Cham, Switzerland: Springer, 2017, pp. 571–593.
[122] L. F. Bittencourt, J. Diaz-Montes, R. Buyya, O. F. Rana, and M. Parashar, "Mobility-aware application scheduling in fog computing," *IEEE Cloud Comput.*, vol. 4, no. 2, pp. 26–35, Mar./Apr. 2017.
[123] E. K. Markakis *et al.*, "EXEGESIS: Extreme edge resource harvesting for a virtualized fog environment," *IEEE Commun. Mag.*, vol. 55, no. 7, pp. 173–179, Jul. 2017.
[124] N. Dragoni *et al.*, "Microservices: Yesterday, today, and tomorrow," in *Present and Ulterior Software Engineering*. Cham, Switzerland: Springer, 2017, pp. 195–216.
[125] M. Jung, S. Mallering, P. Dalbhanjan, P. Chapman, and C. Kassen, "Microservices on AWS," Amazon Web Services, Inc., New York, NY, USA, Tech. Rep., 2016.
[126] B. Butzin, F. Golatowski, and D. Timmermann, "Microservices approach for the internet of things," in *Proc. IEEE 21st Int. Conf. Emerg. Technol. Factory Automat. (ETFA)*, Sep. 2016, pp. 1–6.
[127] T. Vresk and I. Čavrak, "Architecture of an interoperable IoT platform based on microservices," in *Proc. 39th Int. Conv. Inf. Commun. Technol., Electron. Microelectron. (MIPRO)*, May 2016, pp. 1196–1201.
[128] M. S. de Brito *et al.*, "A service orchestration architecture for fog-enabled infrastructures," in *Proc. 2nd Int. Conf. Fog Mobile Edge Comput. (FMEC)*, May 2017, pp. 127–132.
[129] H. Khazaei, H. Bannazadeh, and A. Leon-Garcia, "End-to-end management of IoT applications," in *Proc. IEEE Conf. Netw. Softw. (NetSoft)*, Jul. 2017, pp. 1–3.
[130] L. Sun, Y. Li, and R. A. Memon, "An open IoT framework based on microservices architecture," *China Commun.*, vol. 14, no. 2, pp. 154–162, 2017.
[131] T. Li, Y. Liu, L. Gao, and A. Liu, "A cooperative-based model for smart-sensing tasks in fog computing," *IEEE Access*, vol. 5, pp. 21296–21311, 2017.
[132] P. Krivic, P. Skocir, M. Kusek, and G. Jezic, "Microservices as agents in IoT systems," in *Proc. KES Int. Symp. Agent Multi-Agent Syst., Technol. Appl.*, 2017, pp. 22–31.
[133] M. G. Xavier, M. V. Neves, F. D. Rossi, T. C. Ferreto, T. Lange, and C. A. De Rose, "Performance evaluation of container-based virtualization for high performance computing environments," in *Proc. 21st Euromicro Int. Conf. Parallel, Distrib. Netw.-Based Process. (PDP)*, Feb. 2013, pp. 233–240.
[134] S. Kosta, A. Aucinas, P. Hui, R. Mortier, and X. Zhang, "ThinkAir: Dynamic resource allocation and parallel execution in the cloud for mobile code offloading," in *Proc. IEEE INFOCOM*, Mar. 2012, pp. 945–953.
[135] E. Bastug, M. Bennis, M. Médard, and M. Debbah, "Toward interconnected virtual reality: Opportunities, challenges, and enablers," *IEEE Commun. Mag.*, vol. 55, no. 6, pp. 110–117, Jun. 2017.
[136] Y. Cao, S. Chen, P. Hou, and D. Brown, "FAST: A fog computing assisted distributed analytics system to monitor fall for stroke mitigation," in *Proc. IEEE Int. Conf. Netw., Archit. Storage (NAS)*, Aug. 2015, pp. 2–11.
[137] V. Stantchev, A. Barnawi, S. Ghulam, J. Schubert, and G. Tamm, "Smart items, fog and cloud computing as enablers of servitization in healthcare," *Sensors Transducers*, vol. 185, no. 2, p. 121, 2015.
[138] J. A. Khan, C. Westphal, and Y. Ghamri-Doudane, "Offloading content with self-organizing mobile fogs," in *Proc. 29th Int. Teletraffic Congr. (ITC)*, vol. 1, Jul. 2017, pp. 223–231.
[139] H. Shi, N. Chen, and R. Deters, "Combining mobile and fog computing: Using coap to link mobile device clouds with fog computing," in *Proc. IEEE Int. Conf. Data Sci. Data Intensive Syst. (DSDIS)*, Dec. 2015, pp. 564–571.
[140] T. Wang *et al.*, "Data collection from WSNs to the cloud based on mobile fog elements," *Future Generation Computer Systems*, to be published.
[141] R. Roman, J. Lopez, and M. Mambo, "Mobile edge computing, Fog et al.: A survey and analysis of security threats and challenges," *Future Gener. Comput. Syst.*, vol. 78, pp. 680–698, Jan. 2018.
[142] *OpenFog Reference Architecture for Fog Computing*, OpenFog Consortium Archit. Working Group, Tokyo, Japan, Feb. 2017.

RANESH KUMAR NAHA received the B.Sc. degree in computer science and engineering from the State University of Bangladesh in 2008, the M.Sc. degree from the Department of Communication Technology and Network, Faculty of Computer Science and Information Technology, Universiti Putra Malaysia, in 2015. He is currently pursuing the Ph.D. degree in reliable resource allocation and scheduling in fog computing environment with the University of Tasmania. He has been awarded the Tasmania Graduate Research Scholarship for supporting his studies. He served as a Lecturer with the Daffodil Institute of Information Technology, Bangladesh, until 2011. His research interests include wired and wireless network, parallel and distributed computing, cloud computing, Internet of Things (IoT), and fog computing. During his master's study, he has been awarded the Commonwealth Scholarship provided by the Ministry of Higher Education, Malaysia.

SAURABH GARG is currently a Lecturer with the University of Tasmania, Australia. He is one of the few Ph.D. students who completed in less than three years from the University of Melbourne. He has authored over 40 papers in highly cited journals and conferences. During his Ph.D., he has received various special scholarships for his Ph.D. candidature. His research interests include resource management, scheduling, utility and grid computing, Cloud computing, green computing, wireless networks, and ad hoc networks.

DIMITRIOS GEORGAKOPOULOS is currently a Professor in computer science and the Director of the Key Lab for IoT, Swinburne University of Technology, Melbourne, VIC, Australia. Before that he was the Research Director of the CSIRO's ICT Centre and the Executive Director of the Information Engineering Laboratory, which was the largest Computer Science Program in Australia. Before CSIRO, he held research and management positions at several industrial laboratories in USA, including Telcordia Technologies (where he helped found two of Telcordia's Research Centers in Austin, Texas, and Poznan, Poland); Microelectronics and Computer Corporation (MCC), Austin, TX, USA; GTE (currently Verizon) Laboratories, Boston, MA, USA; and Bell Communications Research, Piscataway, NJ, USA. He was also a Full Professor at RMIT University, and he is currently an Adjunct Professor at Australian National University and a CSIRO Adjunct Fellow. He has produced 170+ journal and conference publications in the areas of IoT, process management, and data management, and has 10 500+ lifetime citations.

PREM PRAKASH JAYARAMAN is a Senior Lecturer and Head of the Digital Innovation Lab in the Department of Computer Science and Software Engineering, Faculty of Science, Engineering and Technology at Swinburne University of Technology. Previously, he was a PostDoctoral Research Scientist at CSIRO. He is broadly interested in emerging areas of Internet of Things (IoT), and Mobile and Cloud Computing. He is the recipient of two best paper awards at HICSS 2017 and IEA/AIE-2010, and contributor to several industry awards including Black Duck Rookie of the Year Award for Open IoT project (www.openiot.eu). He has (co) authored 75+ journal, conference, and book chapter publications that have received 1200+ google scholar citations (h-index: 20), including two seminal papers in IoT (published by Springer) and Industry 4.0 (published by IEEE).

LONGXIANG GAO (SM'17) received the Ph.D. degree in computer science from Deakin University, Australia. He is currently a Lecturer with the School of Information Technology, Deakin University. Before joined Deakin University, he was a Post-Doctoral Research Fellow at IBM Research & Development, Australia. His research interests include data processing, mobile social networks, Fog computing, and network security.

Dr. Gao has over 30 publications, including patent, monograph, book chapter, journal and conference papers. Some of his publications have been published in the top venue, such as IEEE TMC, IEEE IoT, IEEE TDSC, and IEEE TVT. He received the 2012 Chinese Government Award for Outstanding Students Abroad (Ranked No.1 in Victoria and Tasmania consular districts). He is active in the IEEE Communication Society. He has served as the TPC co-chair, a publicity co-chair, an organization chair, and a TPC member for many international conferences.

YONG XIANG received the Ph.D. degree in electrical and electronic engineering from The University of Melbourne, Australia. He is currently a Professor and the Director of the Artificial Intelligence and Image Processing Research Cluster, School of Information Technology, Deakin University, Australia. He has authored or co-authored over two monographs, over 100 refereed journal articles, and numerous conference papers in these areas. His research interests include information security and privacy, signal and image processing, data analytics and machine intelligence, and Internet of Things. He is an Associate Editor of the IEEE SIGNAL PROCESSING LETTERS and the IEEE ACCESS. He has served as a program chair, a TPC chair, a symposium chair, and a session chair for a number of international conferences.

RAJIV RANJAN received the Ph.D. degree in engineering from the University of Melbourne in 2009. He is a Reader at Newcastle University, U.K. Prior to this position, he was a Research Scientist and a Julius Fellow with the CSIRO Computational Informatics Division (formerly known as CSIRO ICT Centre). His expertise is in datacenter cloud computing, application provisioning, and performance optimization. He has authored or co-authored 62 scientific, peer-reviewed papers (seven books, 25 journals, 25 conferences, and five book chapters). His h-index is 20, with a lifetime citation count of 1660+ (Google Scholar). His papers have also received 140+ ISI citations. 70% of his journal papers and 60% of conference papers have been A*/A ranked ERA publication. He has been invited to serve as the Guest Editor for leading distributed systems journals, including the IEEE TRANSACTIONS ON CLOUD COMPUTING, *Future Generation Computing Systems*, and *Software Practice and Experience*. One of his papers was in 2011's Top Computer Science Journal, IEEE COMMUNICATION SURVEYS AND TUTORIALS.