Fog Computing: Platform and Applications

Shanhe Yi, Zijiang Hao, Zhengrui Qin, and Qun Li
Dept of Computer Science, The College of William and Mary
{syi, hebo, zhengrui, liqun}@cs.wm.edu

***Abstract*—Despite the broad utilization of cloud computing, some applications and services still cannot benefit from this popular computing paradigm due to inherent problems of cloud computing such as unacceptable latency, lack of mobility support and location-awareness. As a result, fog computing, has emerged as a promising infrastructure to provide elastic resources at the edge of network. In this paper, we have discussed current definitions of fog computing and similar concepts, and proposed a more comprehensive definition. We also analyzed the goals and challenges in fog computing platform, and presented platform design with several exemplar applications. We finally implemented and evaluated a prototype fog computing platform.**

I. Introduction

The Internet of things (IoT) will be the Internet of future, as we have seen a huge increase in wearable technology, smart grid, smart home/city, smart connected vehicles. International Data Corporation (IDC) has predicted that in the year of 2015, "*the IoT will continue to rapidly expand the traditional IT industry*" up 14% from 2014 [1]. Since smart devices are usually inadequate in computation power, battery, storage and bandwidth, IoT applications and services are usually backed up by strong server ends, which are mostly deployed in the cloud, since cloud computing is considered as a promising solution to deliver services to end users and provide applications with elastic resources at low cost.

However, cloud computing cannot solve all problems due to its own drawbacks. Applications, such as real time gaming, augmented reality and real time streaming, are too latency-sensitive to deploy on cloud. Since data centers of clouds are located near the core network, those applications and services will suffer unacceptable round-trip latency, when data are transmitted from/to end devices to/from the cloud data center through multiple gateways. Besides this, there are also problems unsolved in IoT applications that usually require mobility support, geo-distribution and location-awareness.

The latest trend of computing paradigm is to push elastic resources such as computation and storage to the edge of networks, which motivates the promising computing paradigm of fog computing as a result of prevalence of ubiquitously connected smart devices relying on cloud services. Fog computing keeps data and computation close to end users at the edge of network, and thus provides a new breed of applications and services to end users with low latency, high bandwidth, and location-awareness, and thus gets the name as fog is analogously a cloud close to the ground [2]. We call those facilities or infrastructures providing resources at the edge of the network *fog nodes*. Besides resource-rich servers, fog nodes can be resource-poor devices as well, such as smart TVs/set-top-boxes, gateways, and end devices.

Fig. 1: Three layer architecture:end user/fog/cloud [3]

Fog computing is usually cooperated with cloud computing. As a result, end users, fog and cloud together form a three layer service delivery model, as shown in Fig. 1. Fog computing also shows a strong connection to cloud computing in terms of characterization. For example, elastic resources (computation, storage and networking) are the building blocks of both of them, indicating that most cloud computing technologies can be directly applied to fog computing. However, fog computing has several unique properties that distinguish it from other existing computing architectures. The most important is its close distance to end users. It is vital to keep computing resource at the edge of the network to support latency-sensitive applications and services. Another interesting property is location-awareness; the geo-distributed fog node is able to infer its own location and track end user devices to support mobility. Finally, in the era of big data, fog computing can support edge analytics and stream mining, which can process and reduce data volume at a very early stage, thus cut down delay and save bandwidth.

In the paper, we focus on the fog computing platform design and applications. We will briefly review existing platforms and discuss important requirements and design goals for a standard fog computing platform. We will also introduce some IoT applications to promote the fog computing.

II. Fog Computing Overview

A. Definition

There are a few terms similar to fog computing, such as mobile cloud computing, mobile edge computing, etc. Below we explain each of them.

1) Local Cloud: Local cloud is a cloud built in a local network. It consists of cloud-enabling software running on local servers and mostly supports interplay with remote cloud. Local cloud is complementary to remote cloud by running

978-1-4673-9688-2/15 $31.00 © 2015 IEEE
DOI 10.1109/HotWeb.2015.22

dedicated services locally to enhance the control of data privacy.

2) Cloudlet: Cloudlet is "a data center in a box", which follows cloud computing paradigm in a more concentrated manner and relies on high-volume servers [4]. Cloudlet focuses more on providing services to delay-sensitive, bandwidth-limited applications in vicinity.

3) Mobile Edge Computing: Mobile edge computing [5] is very similar to Cloudlet except that it is primarily located in mobile base stations.

4) Mobile Cloud Computing: Mobile cloud computing (MCC) is an infrastructure where both data storage and data processing happen outside of mobile devices, by outsourcing computations and data storage from mobile phones to cloud [6]. With the trend of pushing cloud to the edge, MCC starts to evolve to mobile edge computing.

5) Fog Computing: Fog computing is generally considered as a non-trivial extension of cloud computing from the core network to the edge network [2]. [7] offers a comprehensive definition of fog computing, which arise from challenges and technologies that will shape the fog, with emphasis on some prominent properties, such as predominance of wireless access, heterogeneity and geographical distribution, sand-boxed environment and flexible interoperability, and large scale of nodes.

However, current definitions are all developed from different perspectives and thus not general. For example, though mobility comes first in edge computing, we do not necessarily narrow it down to mobile edge computing. Fog computing should be defined for a broader range of ubiquitous connected devices. The definition from [7] gives integrative view of fog computing but fails to point out the unique connection to the cloud. We need a more general definition that can abstract all those similar concepts. Here comes our definition: *Fog computing is a geographically distributed computing architecture with a resource pool consists of one or more ubiquitously connected heterogeneous devices (including edge devices) at the edge of network and not exclusively seamlessly backed by cloud services, to collaboratively provide elastic computation, storage and communication (and many other new services and tasks) in isolated environments to a large scale of clients in proximity.*

B. Related Work

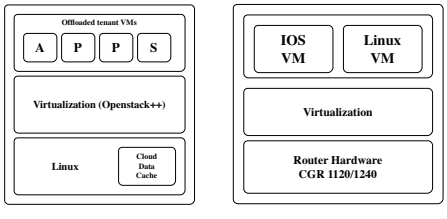
Currently there are a few existing works on the concept of fog computing. Stojmenovic et al. [8], [9] have surveyed [10]–[14]. Our previous work [3], [15] has surveyed additional related work [16]–[21]. In this paper, we further identify work [22], [23] on fog computing.

References [4], [10], [16], [19] are about designing and implementing fog computing nodes. Cloudlet [4], [19] was built before the proposal of fog computing, but inherently coincides fog computing concept. ParaDrop [16] is a fog computing platform implementation based on wireless router, using OS-level virtualization. Hong et al. [10] have proposed a high-level programming model for fog computing platform. References [11], [13], [14], [18], [22], [24] are about how exiting or new applications and services can benefit from fog computing. J. Zhu, et al. [11] have provided dynamic customizable optimization to web applications based on client devices and local network conditions collected by fog nodes. Ha, et al. [24] have designed and implemented a real-time wearable cognitive assistance on Google Glass backed by Cloudlet. Work [18] has designed a cloudlet mesh based intrusion detection system (IDS). Work [13] and [14] are about how rich information collected by fog computing platform can optimize operations or migrations for data processing in fog computing. Cao et al. [22] have explored to use fog computing in health monitoring such as real-time fall detection. Mohammed et al. [23] have conducted an experimental study by utilizing fog computing to assist mobile application in terms of computation offloading and storage expansion.

There are also work related to underlying technologies of fog computing, security and privacy, readers can refer to our previous surveys if interested [3], [15].

III. Design and Implementation

Existing work has focused on the designing and implementing of fog computing nodes such as Cloudlet, IOx [25] and ParaDrop [16]. Cloudlet, as shown in Fig. 2(a), is considered as an exemplar implementation of resource-rich fog nodes. It has a three-layer design, in which the bottom layer is Linux and data cache from cloud, the middle layer is virtualization with a bunch of cloud softwares such as OpenStack [26], and the top layer is applications isolated by different virtual machine (VM) instances. The architecture of IOx is shown in Fig. 2(b), which is a router from Cisco. It works by hosting applications in guest OS running on a hypervisor upon the hardware of a grid router. The platform supports developers to run scripts, compile code, and install their own operation system. This platform is not open to public and relies on expensive hardware. ParaDrop is implemented on gateway (WiFi access point or home set-top box), which is an ideal fog node choice due to its proximity to end user. However, it is designed for home usage scenarios and is not in a fully decentralized manner where all application servers are required to use a ParaDrop Server as entry point to services provided by gateways. We consider this ParaDrop as a complementary implementation of fog computing platform for lightweight task scenarios.



(a) Cloudlet (b) IOx

Fig. 2: Cloudlet architecture (a) adapted from [21] and IOx architecture (b) adapted from [25]

We believe the first step toward fog computing research is the design and implementation of an adequate fog computing platform, which can serve as a standard development environment for prototyping.

A. Designing Goals

There are several designing goals for an adequate fog computing platform.

- Latency. It is fundamental for fog computing platform to offer end user low-latency-guaranteed applications and services. The latency comes from the execution time of a task, the task offloading time, the time for cyber foraging and speed of decisions making, etc.
- Efficiency. While at first glance the efficiency may have its own impact on latency, it is more related to the efficient utilization of resources and energy. The reasons are obvious and quite different from counterparts in cloud computing scenarios: 1) not all fog nodes are resource-rich; some of them have limited computation power, memory and storage. 2) most of fog nodes and clients are battery-powered, such as hand-hold devices, wearables, and wireless sensor units.
- Generality. Due to the heterogeneity of fog node and client, we need provide same abstract to top layer applications and services for fog clients. General application programming interfaces (APIs) should be provided to cope with existing protocols and APIs (e.g. Machine-2-machine protocols, smart vehicle/smart appliance APIs etc).

B. Challenges

It is non-trivial to design fog computing platforms meeting above goals. We could at least identify several challenges ahead.

1) Choice of Virtualization Technology: Virtualization is the main method to provide isolated environments in fog computing and also the main factor of fog node performance. Therefore, an intuitive question here is "hypervisor v.s. container, which one should we choose?" As we know, Cloudlet is utilizing hypervisor virtualization technique while ParaDrop is using a lightweight solution: container i.e. OS-level virtualization. The design choice is made differently since their hardwares have different capabilities. However, one disadvantage against container-based virtualization is the loss of flexibility. For example, it cannot host different kinds of guest operating systems on a single infrastructure node. Therefore, we prefer hypervisor virtualization techniques rather than container-based virtualization.

2) Fight with Latency: There are many factors introducing high latency of application or service performance on fog computing platforms. High latency will ruin the user experience and satisfaction, since fog computing is targeting at delay-sensitive applications and services. There are several possibilities to bring in latency in fog computing:

- Data aggregation. The geo-distributed nature of fog computing paradigm determines that there will be delay if data aggregation is not finished before data processing. However, there are many ways to mitigate this problem, such as applying data partitioning/filtering and utilizing locality in hierarchy to reduce computation volume on higher layer.
- Resource provisioning. There will be delay in provisioning resources for certain tasks, especially for resource-limited fog nodes. We may need carefully designed scheduling by using priority and mobility model.
- Node mobility, churn and failure. Fog computing needs to be resilient to node mobility, churn and failure. Both system monitor and location service will work together to provide information to help on choosing mitigation strategies such as check-pointing, rescheduling and replication.

3) Network Management: The network management will be a burden for fog computing unless we reap the benefits of applying SDN and NFV techniques. However the seamless integration of SDN and NFV into fog computing is not easy, and will certainly be a big challenge. The difficulties come from re-design the south-bound, north-bound and also the east-west-bound APIs to include necessary fog computing primitives. A naive integration may not meet design goals of latency and efficiency.

4) Security and Privacy: We admit that security and privacy should be considered in every stage of fog computing platform design [3]. And we regard it as one of the biggest challenges faced by fog computing. To overcome this, we need apply access control and intrusion detection system, which need support from every layer of the platform.

C. Components

We suggest a fog computing platform be composed of the following components, as shown in Fig. 3. We briefly explain several important components.

1) Authentication and Authorization: The access of fog computing services and resources needs to be authenticated and authorized. One related work [20] has proposed an access control scheme for authorization of heterogeneous resources. Fog computing also opens the door for new authentication and authorization schemes since it is close to end users and can identify user using access pattern, mobility pattern and trusted secure devices.

2) Offloading Management: Offloading is an important component that has impact on all design goals. There are extensive existing research on this topic, as we have surveyed in [15]. The offloading in fog computing needs to solve several problems: 1) what kinds of information are needed in offloading decisions, 2) how to partition application for offloading, and 3) how to design optimal offloading scheme.

3) Location Services: Location services need to maintain a location list of neighbour nodes (mobile and non-mobile node), track mobile end users, and share location information among involved fog nodes. It maps network locations with physical locations, and adopts mobility model provided by end user or learns the mobility model if possible. The tracking and

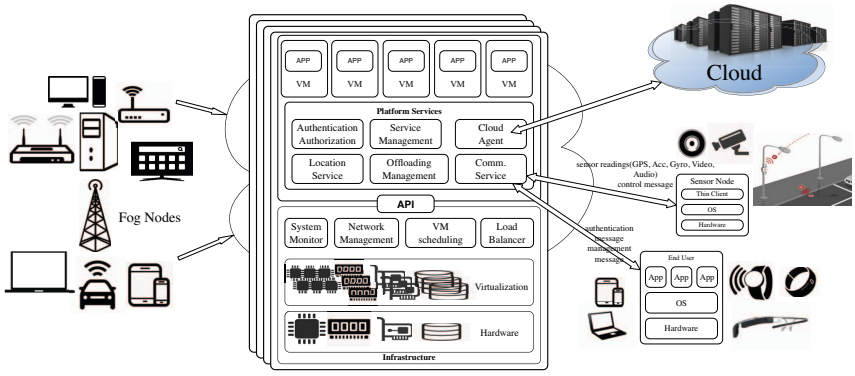


Fig. 3: Components for fog computing platform

mapping on mobile nodes will need information from multiple layers (physical (ultrasound, wireless signal and signature, GPS, IMU sensor), network (IP address), and application (social activities)), which will call for new design for this component.

4) System Monitor: System monitor is a standard component in cloud infrastructures, which can provide useful information such as work load, usage, energy to help lots of decision making and pricing. We highlight this component in fog computing platform since it provides crucial information for other components.

5) Resource Management: The resource management will be responsible for most tasks related to resource discover, resource allocation, the dynamic joining and leaving of fog node, and provisioning and maintaining the resource pool in a distributed manner.

6) VM Scheduling: The VM scheduling needs a brand new design due to fused input of system usage, work load stats, location information and mobility model. New scheduling strategies are needed to provide optimal solution for scheduling VMs.

IV. Applications

The fog computing platform has a broad range of applications. Bonomi et al. [2] have presented fog computing scenarios in connected vehicle, smart grid and wireless sensor and actuator networks (WSAN). Later, Stojmenovic et al. [8], [9] have emphasised previous scenarios and expanded fog computing on smart building. Our previous survey [15] has further highlighted fog computing in augmented reality, content delivery and mobile big data analytics. In this section, we will discuss several fascinating applications that will benefit from fog computing.

A. Smart Home

With the rapid development of the Internet of Things, more and more smart devices and sensors are connected at home. However, products from different vendors are hard to work together. Some tasks, which require large amount of computation and storage, , e.g. real-time video analytics, are infeasible due to the limited capability of hardware. To solve these problems, fog computing is utilized to integrate all debris into a single platform and empower those Smart Home applications with elastic resources.

To use home security application as an example, widely deployed secure sensors consist of smart lock, video/audio recorder, various sensor monitors (e.g. light sensor, occupancy sensor, and motion sensor etc). If not products of same vendor, those secure devices are hard to combine. Fog computing can provide home security applications: 1) unified interface to integrate all kinds of independent devices, 2) flexible resources to support computation and storage, 3) real-time processing and low-latency response. Once the fog platform is set up, each secure sensor is connected as a client. The corresponding server application can be installed in independent VMs. Advanced processing logic can also be implemented on VMs, which can process data shared by those secure monitor applications. For example, a motion sensor detects a suspicious motion in a certain room, then a cleaning robot with video camera will be commanded to check out the exact location. Real-time video analytics will process those video and confirm whether it is a false alarm. Notification and report will be sent to house owner and the system will call police if applicable.

B. Smart Grid

The smart grid is an electricity distribution network, with smart meters deployed at various locations to measure the real-time status information. A centralized server called SCADA system gathers and analyzes the status information, and sends commands to respond to any demand change or emergency to stabilize the power grid.

Fog computing can benefit the smart grid greatly. With fog computing, SCADA can be complemented by a decentralized model with micro-grids, which can not only improve scalability, cost efficiency, security, and rapid response of the power system but also integrate distributed power generators (wind farms, solar panels, and PHEVs) with the main power grid. With fog computing, the smart grid will turn into a multi-tier hierarchical system with the interplay between the fog and SCADA [2], [27]. In such system, a fog is in charge of a micro-grid and communicates with neighbouring fogs and higher tiers. The higher the tier, the larger the latency, and the wider the geographical coverage. The final global coverage

is provided by SCADA, which is responsible for long-time repository and economic analytics.

C. Smart Vehicle

Fog computing can be integrated into vehicular networks. Depending on whether extra infrastructure is needed, vehicular fog computing can be categorized into two types [28], *infrastructure-based* and *autonomous*. The former, such as VTube [29], relies on fog nodes deployed along the roadside; fog nodes are responsible to send/retrieve information to/from the driving-by vehicles. The latter, mentioned in [30], utilizes vehicles on-the-fly to form fog and/or cloud to support ad-hoc events; each fog can communicate its client within and other fogs.

There are various applications for vehicular fog computing, no matter the first type or the second type. Popular applications are: traffic light scheduling, congestion mitigation, precaution sharing, parking facility management, traffic information sharing, etc.

D. Health Data Management

Health data management has been a sensitive issue since health data contains valuable and private information. With fog computing, it is able to realize the goal that patient will take possession of their own health data locally. Those health data will be stored in fog node such as smartphone or smart vehicle. The computation will be outsourced in a private-preserving manner when patient is seeking help from a medical lab or a physicians office. Modification of data happens directly in patient-owned fog node.

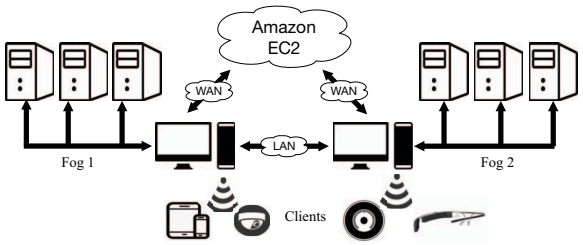


Fig. 4: Testbed setup

V. An Experimental Fog Computing Platform

We build a proof-of-concept fog computing platform, consisting of two fog sub-systems. We install OpenStack on each of them. To be more specific, we install four OpenStack modules: Keystone, Glance, Nova, and Cinder. Keystone is for authentication and authorization; Glance is for VM image management; Nova is a compute module with simple network functionality; and Cinder is the block-level storage module. Note that the two fog sub-systems are two separate OpenStack systems, with a network link between them. To support service continuity, we also implement a VM offloading scheme which can migrate one VM to another fog cluster.

As illustrated in Fig. 4, each of the fog sub-systems possesses one router and three servers. The routers are connected to the Amazon EC2 cloud through WAN, as well as connected with each other through LAN. The routers are also integrated with Wireless AP function, so that mobile devices can access the fog as well as the Amazon EC2 cloud through them.

In the following, we present some preliminary results based on important performance metrics.

A. Latency and Bandwidth

First, we compare the latency and bandwidth provided by fog and cloud. We use RTT (round trip time) as the metric of latency, and we measure both uplink and downlink bandwidth. The results are shown in Table I. We can see that fog computing has strong advantages in terms of low latency and high bandwidth for clients.

TABLE I: Latency and Bandwidth Comparison

	RTT (ms)	Up/Down-link Bandwidth (Mbps)
Fog	1.416	83.723/101.918
Cloud	17.989	1.785/1.746

B. VM Migration

VM migration is essential in fog computing. When a user leaves the area covered by the current fog system, her VMs may need to be migrated to a fog system covering the user's destination. This process should be fast enough to provide satisfactory fog services.

We implement the function of VM migration between the two fog sub-systems shown in Fig. 4, in two different ways. In the first way, Fog 1 takes a snapshot of the VM to be migrated, compresses it, and then transfers the compressed data to Fog 2. Fog 2 then decompresses the data and re-launches the VM by using the snapshot. In the second way, the VM has a "base" snapshot stored on both fogs. The incremental part of the VM's snapshot will be transferred instead of the snapshot itself. To simulate the case that the two fogs are geographically far away from each other, we set their maximum bandwidth to 10 Mbps. The results are presented in Table II.

TABLE II: Fog VM Migration Performance

	Pre-transmission time (sec)	Transmission time (sec)	Post-transmission time (sec)
Full	207.050	143.694	43.982
Incremental	325.116	70.837	63.239

Note that the user will only perceive "Transmission time" + "Post-transmission time" in Table II. "Pre-transmission time" finishes before the user reaches her destination. Therefore, the incremental way is better in our experiments, requiring the user to wait for 134.076 seconds, 53.600 seconds shorter than the full snapshot way. If prediction algorithms are involved, this time could be further reduced, or even eliminated.

C. Face Recognition Application

To further understand the benefits of using fog computing, we implement a face recognition application running across a smartphone and a fog or a cloud. The application requires to install an app on the user's smartphone. The user can use the

app to capture a face photo of herself or any other people, and the app will transmit the face photo to a remote server, either in a fog or in a cloud. The remote server will then try to recognize the face by matching it in the local face photo database. In our implementation, the app running on the smartphone takes photos of 384x286 pixels. The face photo database on the remote server consists of 1521 photos, each of which is also 384x286 pixels.

We run the same tasks on our fog as well as on the Amazon EC2 cloud. Table III presents the results.

TABLE III: Fog-based Face Recognition Performance

	Face recognition time on the server (ms)	Response Time (ms)
Fog	2.479	168.769
Cloud	2.492	899.970

"Response Time" in Table III is the time duration from when the smartphone begins to upload the face photo to when the smartphone receives the result from the remote server. Providing the similar VM capabilities, the fog and the cloud consume similar time on the computation task (about 2.5 ms in our case). From Table I, we know that the difference of latency between the fog and cloud is small (less than 10 ms). Therefore, the network bandwidth contributes the most to the big difference in response time (731.201 ms). If we assume the smartphone can upload the face photo in negligible time to the fog, then the smartphone can upload about 167 KB data to the cloud in this 731.201 ms, which is comparable to the average size of uploaded face photos in our experiments (about 110 KB).

VI. Conclusion and Future Work

We have briefly introduced fog computing and given a more comprehensive definition of fog computing after analysing similar concepts. In the paper, we have discussed the design goals and challenges of a fog platform. We have presented the design and implementation of a prototyping platform for fog computing. In the end, we have evaluated our prototyping platform in Smart Home applications. Future work includes full-fledged fog platform implementations and performance optimizations.

References

[1] Gil Press, "Idc: Top 10 technology predictions for 2015," http://goo.gl/zFujnE, 2014, [Online; accessed 18-June-2015].

[2] F. Bonomi, R. Milito, J. Zhu, and S. Addepalli, "Fog computing and its role in the internet of things," in *workshop on Mobile cloud computing*. ACM, 2012.

[3] S. Yi, Z. Qin, and Q. Li, "Security and privacy issues of fog computing: A survey," in *International Conference on Wireless Algorithms, Systems and Applications (WASA)*, 2015.

[4] M. Satyanarayanan, P. Bahl, R. Caceres, and N. Davies, "The case for vm-based cloudlets in mobile computing," *Pervasive Computing*, 2009.

[5] ETSI, "Mobile-edge computing," http://goo.gl/7NwTLE, 2014, [Online; accessed 18-June-2015].

[6] H. T. Dinh, C. Lee, D. Niyato, and P. Wang, "A survey of mobile cloud computing: architecture, applications, and approaches," *WCMC*, 2013.

[7] L. M. Vaquero and L. Rodero-Merino, "Finding your way in the fog: Towards a comprehensive definition of fog computing," *ACM SIGCOMM CCR*, 2014.

[8] I. Stojmenovic, "Fog computing: A cloud to the ground support for smart things and machine-to-machine networks," in *Telecommunication Networks and Applications Conference (ATNAC)*. IEEE, 2014.

[9] I. Stojmenovic and S. Wen, "The fog computing paradigm: Scenarios and security issues," in *Federated Conference on Computer Science and Information Systems (FedCSIS)*. IEEE, 2014.

[10] 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 *ACM SIGCOMM workshop on Mobile cloud computing*, 2013.

[11] J. Zhu *et al.*, "Improving web sites performance using edge servers in fog computing architecture," in *SOSE*. IEEE, 2013.

[12] H. Madsen, G. Albeanu, B. Burtschy, and F. Popentiu-Vladicescu, "Reliability in the utility computing era: Towards reliable fog computing," in *IEEE International Conference on Systems, Signals and Image Processing (IWSSIP)*, 2013.

[13] K. Hong, D. Lillethun, U. Ramachandran, B. Ottenwälder, and B. Koldehofe, "Opportunistic spatio-temporal event processing for mobile situation awareness," in *Proceedings of the ACM international conference on Distributed event-based systems*, 2013.

[14] B. Ottenwälder, B. Koldehofe, K. Rothermel, and U. Ramachandran, "Migcep: operator migration for mobility driven distributed complex event processing," in *Proceedings of the ACM international conference on Distributed event-based systems*, 2013.

[15] S. Yi, C. Li, and Q. Li, "A survey of fog computing: Concepts, applications and issues," in *Proceedings of the 2015 Workshop on Mobile Big Data*. ACM, 2015.

[16] D. F. Willis, A. Dasgupta, and S. Banerjee, "Paradrop: a multi-tenant platform for dynamically installed third party services on home gateways," in *ACM SIGCOMM workshop on Distributed cloud computing*, 2014.

[17] J. K. Zao *et al.*, "Pervasive brain monitoring and data sharing based on multi-tier distributed computing and linked data technology," *Frontiers in human neuroscience*, 2014.

[18] Y. Shi, S. Abhilash, and K. Hwang, "Cloudlet mesh for securing mobile clouds from intrusions and network attacks," in *The Third IEEE International Conference on Mobile Cloud Computing, Services, and Engineering*, 2015.

[19] M. Satyanarayanan, Z. Chen, K. Ha, W. Hu, W. Richter, and P. Pillai, "Cloudlets: at the leading edge of mobile-cloud convergence," in *IEEE International Conference on Mobile Computing, Applications and Services (MobiCASE)*, 2014.

[20] C. Dsouza, G.-J. Ahn, and M. Taguinod, "Policy-driven security management for fog computing: Preliminary framework and a case study," in *IEEE International Conference on Information Reuse and Integration (IRI)*, 2014.

[21] M. Satyanarayanan, R. Schuster, M. Ebling, G. Fettweis, H. Flinck, K. Joshi, and K. Sabnani, "An open ecosystem for mobile-cloud convergence," *IEEE Communications Magazine*, 2015.

[22] Y. Cao, P. Hou, D. Brown, J. Wang, and S. Chen, "Distributed analytics and edge intelligence: Pervasive health monitoring at the era of fog computing," in *Proceedings of the 2015 Workshop on Mobile Big Data*. ACM, 2015.

[23] M. A. Hassan, M. Xiao, Q. Wei, and S. Chen, "Help your mobile applications with fog computing," in *Fog Networking for 5G and IoT Workshop*, 2015.

[24] K. Ha, Z. Chen, W. Hu, W. Richter, P. Pillai, and M. Satyanarayanan, "Towards wearable cognitive assistance," in *Mobisys*. ACM, 2014.

[25] Cisco, "Iox overview," http://goo.gl/n2mfiw, 2014, [Online; accessed 18-June-2015].

[26] Openstack, "Openstack open source cloud computing software," https://www.openstack.org/, 2015, [Online; accessed 18-June-2015].

[27] C. Wei, Z. M. Fadlullah, N. Kato, and I. Stojmenovic, "On optimally reducing power loss in micro-grids with power storage devices," *IEEE Journal on Selected Areas in Communications*, 2014.

[28] G. Yan, D. Wen, S. Olariu, and M. C. Weigle, "Security challenges in vehicular cloud computing," *IEEE Transactions on Intelligent Transportation Systems*, 2013.

[29] T. H. Luan, L. X. Cai, J. Chen, X. Shen, and F. Bai, "Vtube: Towards the media rich city life with autonomous vehicular content distribution," in *IEEE Communications Society Conference on Sensor, Mesh and Ad Hoc Communications and Networks (SECON)*, 2011.

[30] M. Eltoweissy, S. Olariu, and M. Younis, "Towards autonomous vehicular clouds," in *Ad hoc networks*. Springer, 2010.