IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received December 30, 2017, accepted January 26, 2018, date of publication February 7, 2018, date of current version March 16, 2018.

Digital Object Identifier 10.1109/ACCESS.2018.2803439

Fog-Based Crime-Assistance in Smart IoT Transportation System

AUGUSTO J. V. NETO[1,2], ZHONGLIANG ZHAO[3], JOEL J. P. C. RODRIGUES[2,4,5,6], (Senior Member, IEEE), HUGO BARROS CAMBOIM[1,2], AND TORSTEN BRAUN[3]

[1]Federal University of Rio Grande do Norte, Natal 59078-970, Brazil
[2]Instituto de Telecomunicações, 1049-001 Lisbon, Portugal
[3]University of Bern, 3012 Bern, Switzerland
[4]National Institute of Telecommunications (Inatel), Santa Rita do Sapucaí 37540-000, Brazil
[5]ITMO University, 197101 Saint Petersburg, Russia
[6]University of Fortaleza, Fortaleza 60811-905, Brazil

Corresponding author: Augusto J. V. Neto (augusto@dimap.ufrn.br)

This work was supported in part by the National Funding from the CNPq through the SMART (MCTI/CNPQ/Universal 14/2014) project in the scope of the Research Group in Future Internet Services and Applications (REGINA) under Grant 457051/2014-0, in part by the National Funding from the FCT - Fundação para a Ciência e a Tecnologia under Grant UID/EEA/50008/2013, in part by the Government of the Russian Federation under Grant 074-U01, in part by Brazilian National Council for Research and Development (CNPq) under Grant 309335/2017-5, in part by by Finep with resources from Funttel through the Centro de Referência em Radiocomunicações - CRR project of the Instituto Nacional de Telecomunicações (Inatel), Brazil, under Grant 01.14.0231.00.

ABSTRACT Smart transportation safety (STS) envisions improving public safety through a significant paradigm shift for police authority responses on crimes toward a pro-active one. The application of smart surveillance in STS is critical for automatic and accurate identification of events in case of security threats in target environments. Cloud computing reduces costs and high resource consumption of smart surveillance capable STS systems, at the cost of introducing additional latency through far away centralized systems. In this paper, the fog-framework for intelligent public safety in vehicular environment (FISVER) framework applies fog computing in smart video surveillance-based STS to enhance crime assistance in a cost-efficient way. Through fog-FISVER, in-vehicle and fog infrastructures support autonomous and real-time crime detection on public bus services. A fog-FISVER laboratory testbed prototype was created and extensive evaluations in a real testbed were performed. Results show that fog-FISVER delivers outstanding system performance and device survivability behavior over typical STS use cases.

INDEX TERMS Smart transportation safety, fog computing, smart video surveillance, ubiquitous computing, Internet of Things (IoT).

I. INTRODUCTION

Wireless networking, mobile computing, seamless communications, cloud computing, ubiquitous/pervasive computing, etc., are key technological enablers for innovating services and applications that are supporting traditional cities to become smart. Smart Transportation Safety (STS) in Smart Cities is attracting particular attention from both research and industry communities because of the sharp rise in incidents involving vehicle thefts as well as violent crimes on urban buses. Authorities often respond to those incidents in a rather inefficient way, especially in developing countries. Traditionally, police agents respond to crimes reactively (after-the-fact), mostly relying on the population engagement to report crime incidents. Incident response police vehicle stands for the car that police agents use to deal with a reported crime. Apart from that, crime details are usually obtained only after the incident occurs, through offline analysis in video feeds compiled by on-bus cameras. Therefore, such reactive scheme is naturally not effective to avoid incidents. Furthermore, there is a high risk for human failures in stressful conditions. STS systems enable a shift in the paradigm that today's police authority adopts to respond reported crimes, from a reactive to a predictive model. The goal of the predictive model is to anticipate procedures to be taken before the fact in attempt to avoid crime incidents on public buses, and reducing fatalities.

STS can be defined as a set of systems that feature computing capabilities for collecting, processing, and analyzing data

to identify potential security threat events (e.g., robberies, fights, accidents, kidnappings, etc.) in public transportation [1]. From the analysis of these massive amounts of continuously updated information, STS systems allow discovery of unknown patterns, trends, and correlations in order to detect, predict, and prevent crime and security threats with short latency. This will allow better quality of life, customer service, improved operational efficiency, and increased revenues. Tools and technologies enabling such massive data collection and processing are provided by STS *cloudification*, while data analysis is supported by Big Data analytics.

Smart video surveillance [2] is a key enabler for STS and enables analytics of collected video data in real-time to automatically identify events of potential security threats in the target environment [3]. However, STS based on smart video surveillance requires a tremendous amount of processing, storage, and networking resources for multimedia data transmissions to remote systems that will perform real-time analytics. Such infrastructures are very costly because of the demand for high-performance, robust, and reliable servers to host particular service applications. In this context, the *cloudification* of STS systems plays a vital role by provisioning high- performance resources for complex and resource-consuming video processing and analytics tasks.

Addressing the primary issues of STS requirements [4] (e.g., scalability, ubiquitous sensory data access, event processing overhead, high networking traffic, and massive storage), fog computing [5] emerges as a promising alternative. Fog computing aims at efficiently supporting latency-critical STS applications, by providing cloud computing functionality at nodes in proximity to data producers. Fog computing aims to support high responsiveness in STS systems by eliminating long round-trip times introduced by far away centralized cloud infrastructures used for analytics. Processing data at fog nodes will also drastically reduce costs, save energy, and reduce bandwidth consumption [6].

In a previous work, the authors proposed the FISVER (Framework for Intelligent Public Safety in Vehicular Environment) [7] that applies STS-tailored cloud services to support crime detection, prediction, and prevention. For this work, the FISVER framework is adapted to apply ultimate ICT paradigms to advance towards a highly enhanced STS system architecture. The resulting approach, called Fog-FISVER, provides a 3-tier architectural framework that orchestrates in-vehicle and fog systems to address enhancing public security in Smart Cities by autonomous and real-time crime detection at bus transportation services. Fog-FISVER envisions assisting police authorities in finding the location of a crime event and thus getting the ability to respond to the event quickly. Fog-FISVER's main contributions are the following: (*i*) an in-vehicle smart video surveillance fog subsystem with capacities to detect occurrence of criminal threats in real-time; *(ii)* a fog-enabled infrastructure, acting as a front-end between target vehicles and incident response police vehicles; and *(iii)* event-driven mobile application, that foresees to report police agents close enough to deal with a detected crime.

The rest of the paper is organized as follows. Section II discusses most relevant related works. Section III provides a detailed description of the proposed Fog-FISVER framework. Evaluation results and discussion of the Fog-FISVER laboratory prototype are described in Section IV. Finally, Section V concludes the paper and gives an outlook to future work.

II. RELATED WORK

This section surveys relevant works on STS-tailored service applications leveraging smart surveillance and analysis techniques. Proposals leveraging cloud-based video surveillance [8]–[11] operate by continuously streaming data of video surveillance cameras to far away cloud infrastructures for further analytics. All these works need broadband connectivity to allow video streaming with high quality. Otherwise, the accuracy in detection and recognition tasks running in the cloud cannot be assured. To overcome the challenges raised above, some works [12], [13] attempt to reduce bandwidth by video quality adaptations, whereas the system presented in [14] applies video compression for streaming. All these proposals are unsuitable for latency-critical use cases, since the additional latency caused by applying video optimization technologies (e.g., compression, transcoding, or re-buffering events) in attempt to reduce bandwidth-consume rates, affects the video quality.

The feasibility of applying fog computing for public surveillance applications has been investigated [15], [16]. In these works, fog computing is used for processing tasks of tracking, speed monitoring, and detection of event incidents that occur in squares and streets. Industrial solutions require low-cost cameras that enable powerful image processing schemes, which trigger instantaneous alerts by email and/or text messages in case of any detected pattern change. Examples are Netatmo [17], Skywatch [18], Brickcom [19], and Smartvue [20]. Even though these industrial solutions are beneficial to detect on-going critical security threats (e.g., intruders), sometimes, it becomes a hassle when the notifications are continuously sent for less important reasons (e.g., detecting butterflies or pets), which makes systems inefficient.

For the design of an efficient smart surveillance STS system, the following crucial principles are important: (*i*) Agile and accurate detection and classification of crime events by in-vehicle devices in real-time; *(ii)* Service deployment at fog nodes to ensure high-performance to support latency-critical applications; *(iii)* Integration with municipality backend systems for enhanced city-level computing; and *(iv)* Crime detection and prediction units relying on mobile computing for agile and customized triggering of events. It becomes evident that none of the related works can meet all these critical application requirements for the reasons described in the following. Firstly, most smart surveillance STS solutions provide storage and real-time video streaming as cloud

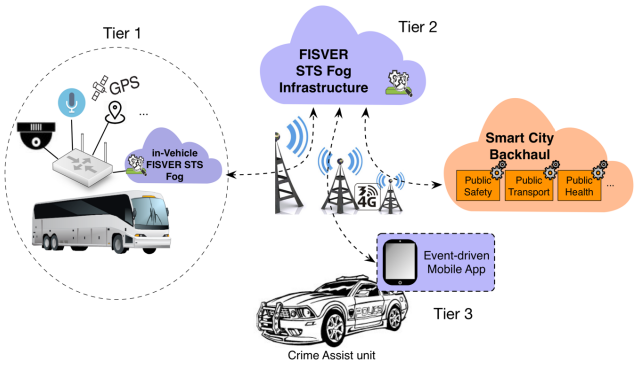

FIGURE 1. Illustration of a high-level view in a Fog-FISVER enabled ecosystem.

services, thus potentially resulting in low rates of agility (by high latency), survivability (consuming considerable energy) and scalability (severe networking costs). Moreover, using carrier-grade multimedia networking is very costly, especially when it is necessary to afford minimum quality guarantees (which is the case to ensure accurate video analytics).

III. FOG-FISVER FRAMEWORK

Fog-FISVER aims to provide a new fog-capable (rather than the high-latent and costly cloud-based approach) architectural framework for the efficient support of smart video surveillance based STS applications that detect and predict crime incidents quickly and automatically, in the meanwhile keeping a low network overhead. In the next subsections, design details of the Fog-FISVER architecture are presented.

A. FOG-FISVER KEY DESIGN PRINCIPLES

Fog-FISVER notably differentiates itself from our previous work [7], as well as from the related works described in Section II, through designing its framework following the fog-computing paradigm principles, with the prospect to achieve enhanced performance impact. In general terms, fog-computing addresses the network edge by defining both video data gathering and real-time analytics to efficiently satisfy low latency requirements of video-processing based STS through quick access and analysis of sensory data locally at the in-vehicle fog node. In contrast, our previous work is based on a centralized architecture at a data center infrastructure to provide crime threat detections as a cloud service, in which the in-vehicle video camera must continuously stream the video data to the cloud for analysis. Fog-FISVER features a 3-tier approach, as shown in Figure 1.

The first tier refers to an in-vehicle fog node featuring mechanisms for fetching local sensory data and implementing local-level crime analytics. The second tier is running at the fog computing infrastructure, performing high-performance crime event analytics for threat incidence classification and confirmation. The second tier also tries to find incident response police vehicles that best meet capacities to quickly deal with a detected crime event. Lastly, the third tier refers to mobile applications running on mobile devices at incident response police vehicles, which in turn reports a crime to the end user (the police agent) within a short latency.

The Fog-FISVER conceptual framework design follows a modular approach that features three interworking subsystems accessible via well-defined web-based interfaces. In the modular architecture of the Fog-FISVER framework, each subsystem has clear divisions among subcomponents, which can be replaced/updated without affecting the rest of the system. Each subsystem has its own architecture, and subcomponents communicate with each other via well-known internal interfaces. Figure 2 sketches a general workflow diagram that shows the resulting schematic framework when a smart surveillance based STS system design complies with the Fog-FISVER conceptual framework.

The Fog-FISVER conceptual architecture is presented in Figure 3, depicting all components and subcomponents, as well as internal and external accessibility interfaces. Afterwards, the key Fog-FISVER capable system component recommendations are discussed, including all sequence of operations complying with the workflow diagram that the Figure 2 sketches.

1) IN-VEHICLE FISVER STS FOG COMPONENT

Inside the vehicle, a fog node hosts the in-Vehicle FISVER STS Fog component, which includes a set of subsystems to detect local crime threats in real-time. The main functionalities are as follows: (i) gathering sensory data in the bus cabin; (ii) processing multimedia sensory data to identify potential security threats; (iii) creating crime-level metadata; and (iv) triggering the FISVER STS Fog Infrastructure to provide crime-level metadata.

a: IMAGES PROCESSOR

Inside the architecture of the in-Vehicle FISVER STS Fog component, the Images Processor module must be designed as a computer vision system, that brings the idea to duplicate the human vision ability. Therefore, the Images Processor module design must follow digital image processing technologies to provide deep learning-based real-time crime object detector capabilities.

In our Fog-FISVER framework laboratory prototype, the in-Vehicle FISVER STS Fog component is implemented in a Raspberry Pi 3 model B (1.2GHz Quad-Core ARMv8 CPU, 1 GB RAM, and 64GB flash disk card) minicomputer that is equipped with a high-quality video camera. The deep learning-based real-time object detection capability that the Images Processor brings, is implemented in Java (http://java.com) with the aid of the highly efficient Open Source Computer Vision (OpenCV) v3.1.0 (http://opencv.org). The *VideoStream* class facility, which is the most used library for object detection and tracking, is applied to understand the surrounding world captured by image sensors.

We highlight that this paper focuses on the Fog-FISVER framework, and thus it is out of the scope to define detailed

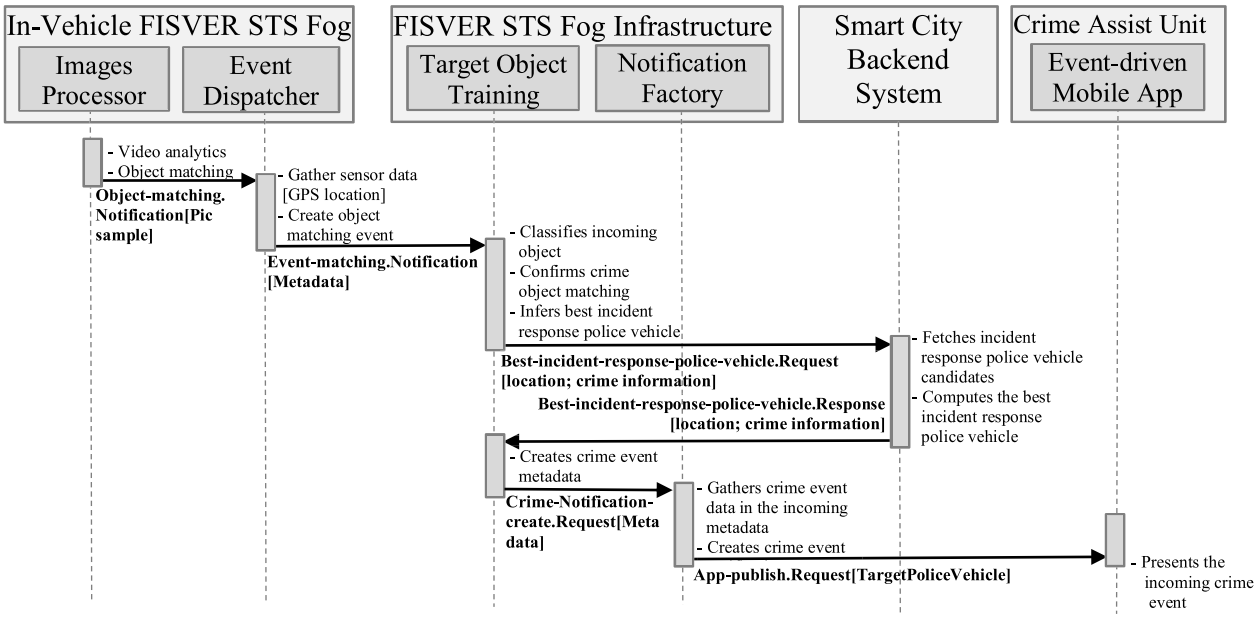

FIGURE 2. General workflow diagram of the Fog-FISVER conceptual framework proposal.

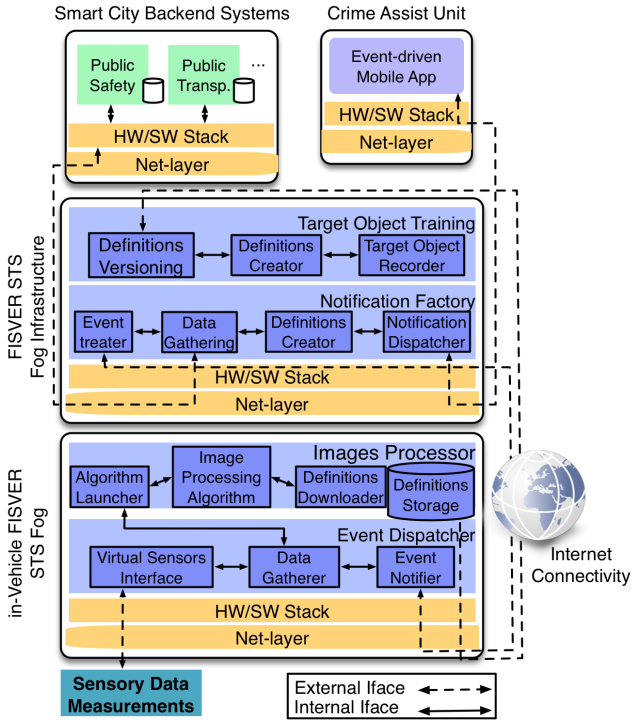

FIGURE 3. Illustration of a high-level view in a Fog-FISVER enabled ecosystem.

hardware- and software-type tools and approaches, all were exclusively set to afford the laboratory prototyping. The Fog-FISVER holistic architectural framework is proposed to guide the development of Smart Surveillance based STS, with performance-enhanced system perspectives. In light of this, we stand that application and system engineering team is in

FIGURE 4. Real-time crime object recognition in video feed by Images Processor laboratory prototype [7].

charge of the elicitation of the underlying system requirements to further define the proper software/hardware platform, leveraging a number of options available in the market for this.

The Images Processor leverages the Template Matching and Correlation Method [21] for high accurate computer vision techniques for object detection. In the scope of the Fog-FISVER framework, the Images Processor analyzes whether an input video feed contains a sub-image that matches dangerous artifacts regularly utilized in criminal incidents at buses (i.e., a crime object template image, e.g., pistol or a knife). For creating template definitions and implementing image analytics, *Haar* features [25] are used to implement the Image Processor laboratory prototype. Figure 4 shows the Image Processor laboratory prototype when detecting an input video feed containing a crime object.

One instance of the image processing algorithm is assigned to deal with one crime object template individually. The key subcomponents of the Images Processor module are the following:

- Crime definitions downloader: This component periodically verifies if a new version of the crime object template definition is available at the Fog-FISVER STS Fog Infrastructure. If so, the crime object template definitions are downloaded and stored locally;
- Crime definitions storage: This component is responsible for storing all crime object template definitions, and delivering them on query actions from the algorithm launcher;
- Algorithm launcher: After all crime object template definitons have been downloaded and stored, the algorithm launcher creates a new instance of each registered algorithm, passing the specific configuration file that contains parameters for one particular crime object template. Each algorithm instance monitors video feeds and searches for a crime object template match. Whenever matching a crime object template in a video feed, it stands for an achieved crime detection.

b: EVENT DISPATCHER SUBCOMPONENT

The Event Dispatcher module must be designed to aggregate information from both the Images Processor and the intra-vehicular sensor units. Once a crime event has been detected, this information is grouped in another XML type file, which is prepared and sent towards the FISVER STS Fog Infrastructure for second-level processing. The subcomponents of the Event Dispatcher are as follows:

- The Event Notifier is responsible for collecting data extracted from the registered array of sensors and sends them to the FISVER STS Fog Infrastructure for processing. GPS location, vehicle temperature, toxic gas, fire, and others are examples of in-vehicle sensors.
- The Data Gatherer intermediates operations between the Event Notifier and sensors as well as abstracts interfaces for gathering data from registered sensors.
- The Virtual Sensors Interface implements a virtual layer for data sensing. The sensory data can be collected from different sources, and each one must follow specific procedures complying with the target sensor platform and protocols.

The operation of the Event Dispatcher can be summarized as follows: whenever an algorithm instance matches a given video with a crime object template, the Event Notifier is invoked, which then triggers the Data Gatherer for providing raw sensory data. After receiving the sensory data from the Data Gatherer, the Event Notifier composes an event XML file that includes crime-level metadata. An image sample containing the object of the detected crime, along with raw sensory data and fills the crime-level metadata. Lastly, the Event Notifier delivers the XML file towards the FISVER STS Fog Infrastructure through a Web service. Listing 1 shows an example of the XML file to report an event incidence by the Event Notifier:

The Virtual Sensors Interface provides functions for both registering and gathering raw sensory data to feed the Data

```
<?xml version="1.0"?>
<event>
  <!-Potential threat event instance-!>
 <bus id="9999">
  <location>
   <lat>5.840592</lat>
   <long>-35.1999164</long>
  <location>
  <company id="45">
   <name>Example Company</name>
   <line>45</line>
 </bus>
 <avg_speed>40</avg_speed>
 <fire>false</fire>
 <algorithm id="1001"
 name="cascade_classifier">
  <image_res>640_480</image_res>
  <detectime></detectime>
 </algorithm>
 <image_cut>
  R0lGODlhPQBEAPeoAJosM//AwO/AwHVYZ/z595kzAP/s7P+go
  OXMv8+
 </image_cut>
</event>
```

Listing 1. Example of a XML file featuring crime threat metadata.

```
<?xml. version="1.0"?>
<!-configuration file for a target
object detection-!>
<storage>
 <cascade type_id="cascade-classifier">
  <stageType>BOOST</stageType>
  <featureType>HAAR</featureType>
   <height>20</height>
   <width>20</width>
  <stageParams> <maxWeakCount>213
  </maxWeakCount> </stageParams>
  <featureParams> <maxCatCount>0
  </maxCatCount> </featureParams>
  <stageNum>22</stageNum>
  <stages>
   <maxWeakCount>3</maxWeakCount>
     <stageThreshold>8.226894140243530e-01
   </stageThreshold>
     <weakClassifiers>
      <internalNodes> O -1 0 4.0141958743333817e-03</internalNodes>
      <leafValues> 3.3794190734624863e-02 8.3781069517135620e-01
   </leafValues>
  </weakClassifiers>
  </stages>
</storage>
```

Listing 2. Example of a target object definition file.

Gatherer with details of sensor measurements. This knowledge is provided during the registering phase.

2) FISVER STS FOG INFRASTRUCTURE

The STS Fog Infrastructure embeds the Fog-FISVER subsystems running on top of the Fog computing substrate. The services and applications of the FISVER STS Fog Infrastructure are summarized as follows: (*i*) Templates as a Service, keeping in-vehicle algorithms always up to date with new crime object template definitions; *(ii)* Event Classification as a Service using intelligent computing algorithms for classification of events in real-time based on the analysis of the images sent by the Vehicle-side STS Fog (i.e., Event Dispatcher); and

(iii) Incident Response Triggering as a Service to find the best-suited police vehicle and further sending crime reports.

A. TRAINING OF TARGET OBJECT RECOGNITION

Since there is a wide range of crime objects, the Target Object Training Module makes it possible to add new objects to be recognized. These can only be accessed by the police authorities (i.e. a security analyst), which can insert, update, or delete crime object template definitions. The main tasks of the Target Object Training component are creating, updating, and storing crime object template definitions for updating the in-Vehicle STS Fog instances, in which the Images Processor subcomponent uses them through the Algorithm Launcher as a parameter for crime object template matching.

The proposed approach creates a file that includes template definitions about crime object templates (e.g., color gradient, shape format, etc.). The template definitions include a set of samples that correspond to images featuring crime object(s). The system administrator is responsible to provide crime object template datasets, as well as keeping the template definitions always up to date to support the in-vehicle Subsystems for accurate crime incident detection. To achieve this, the system administrator must execute a training step, which will output an XML file containing definitions of a crime object. Figure 4 shows an example of a template definition XML file created after a training step.

On completing the crime object template definition creation process, the Definitions Versioning subcomponent creates a new entry for a new crime object template definition, and stores it in the Definitions Database. On updating an existing template definition with new object patterns, a new version identification code is set. The in-Vehicle STS Fog Subsystem checks periodically in the FISVER STS Fog Infrastructure whether a new version of template definitions is available.

B. NOTIFICATION FACTORY

The Notification Factory subcomponent is responsible for receiving event notifications sent by all in-Vehicle STS Fog Subsystem instances. Once a threat event has been dispatched from the in-Vehicle STS Fog Subsystems, the Event Treater subcomponent validates the received event. In order to guarantee FISVER STS Fog Subsystems service continuity, the Event Treater should work with a processing queue, in which incoming events are received and queued for a validation process that discards wrong event messages.

C. EVENT-DRIVEN MOBILE APPLICATIONS

The Event-driven Mobile Application runs on end user devices (e.g., smartphones/tablets at incident response police vehicles, third party safety centers, police authority control centers, etc.) deploying lightweight procedures, mostly for receiving crime incident notifications and presenting them to the police agent. To support this, the Event-driven Mobile Application must follow a notification-based approach for staying in standby mode until receiving crime notifications from the FISVER STS Fog Infrastructure subsystems.

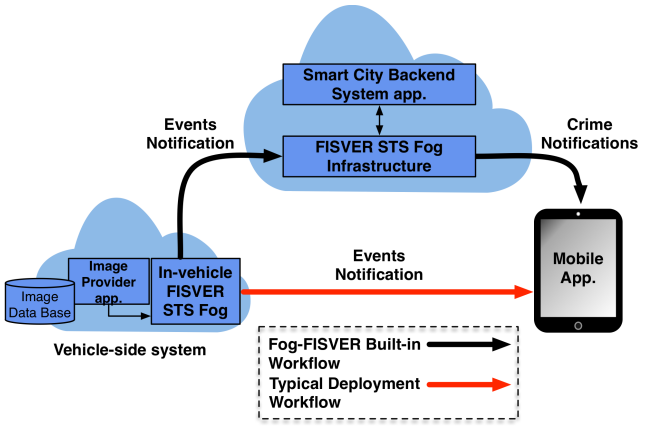

FIGURE 5. Testbed configurations used in the evaluations for both Typical Deployment and Fog-FISVER Built-in experiments.

Given the benefits expected from the components of the Fog-FISVER framework, a broad range of applications for smart public security can be designed. The mobile App, exclusively designed for implementing the Event-driven Mobile Application in the Fog-FISVER laboratory prototype, was created. A Typical NDK scenario (https://developer.android.com/ndk) was used to make native library loading for the Android mobile App, which are made through JNI.

IV. PERFORMANCE EVALUATION OF THE FOG-FISVER FRAMEWORK

In this section, the suitability and feasibility of Fog-FISVER to efficiently support Smart Surveillance based STS systems by deploying a software architecture with scalable, robust, and computationally efficient capabilities are discussed. A laboratory testbed prototyping a Smart Surveillance based STS use case that complies with the Fog-FISVER conceptual framework, has been deployed. Considering that Smart Surveillance based STS systems impose stronger requirements than in usual surveillance applications (as raised in Section II), benchmarking of computation resource consumption in different use cases is required. Following the guidelines of related work [22], [23], two laboratory testbed experiments have been adopted, denoted as Typical Deployment and the Fog-FISVER Built-in. Figure 5 sketches the laboratory testbed configuration used in the evaluations for both Typical Deployment and Fog-FISVER Built-in experiments.

Both the Image Provider application and the In-Vehicle FISVER STS Fog subsystem are running at the incident response police vehicle. The Image Provider application fetches images periodically and immediately delivers those to the In-Vehicle FISVER STS fog subsystem, which carries out real-time local processing and analysis. A set of images that strategically represent crime events were generated. To do this, each image features a different crime object with the goal to drive the In-Vehicle FISVER STS fog subsystem for matching the crime object template in a controllable way. This scheme allows simulating the behavior of smart video

TABLE 1. Simulation parameter used in the experiments.

Simulation Parameter	Images (unit)	Image Fetching Time (s)	Crime Objects (unit)	Experiment Time (minutes)
Amount	20	Random	4	10

surveillance devices, thus allowing implementing different STS use cases through a controllable evaluation environment. The simulation parameters are summarized in Table 1.

On one side, the implementation of the Typical Deployment set of experiments follows the guidelines of [24] for raising configuration sets typically used in smart video surveillance assessments for STS (as those highlighted in Section II). The corresponding workflow of the Typical Deployment laboratory testbed configuration is as follows: on matching a crime object template at the image provided by the Image Provider application, the In-Vehicle FISVER STS Fog component delivers an event notification (carrying the crime image along with corresponding metadata) to the Event-driven Mobile Application. At the Mobile Application side, a second-level video analytics component is deployed for crime confirmation and classification. On matching a confirmation, the user is notified on the mobile device screen accordingly.

On the other side, the Fog-FISVER Built-in experiments aim at prototyping a smart video surveillance based STS use case that follows the Fog-FISVER architectural framework as a whole. Similarly to the Typical Deployment testbed set, the Fog-FISVER Built-in experiments deploy the same vehicle- side system approach (i.e., the Image Provider application interworking with the In-Vehicle FISVER STS fog subsystem), featuring the workflow described as follows: on matching a crime object, the In-Vehicle FISVER STS Fog triggers event notifications towards the FISVER STS Fog Infrastructure, which in turn classifies the indicated event. To confirm a crime event, the FISVER STS Fog Infrastructure uses the derived GPS location to infer, in an accessible backend system, the best incident response police vehicle. An application called Smart City Backend system was created. It runs locally on the FISVER STS Fog Infrastructure to keep a data set featuring different incident response police vehicle locations (location set randomly by the authors, and maintained statically for simplified assessments effort in the Fog-FISVER framework). The Smart City backend system application implements a Web service to provide an incident response police vehicle that best deals with a given crime incident, namely, the police vehicle that is closer to the crime incident location. Upon fetching the best incident response police vehicle, the FISVER STS Fog Infrastructure extends the corresponding metadata with the crime event classification and, then, delivers it to the selected incident response police vehicle.

A. METHODOLOGY USED FOR THE PERFORMANCE EVALUATION

In order to evaluate the suitability, feasibility, and performance of the proposed Fog-FISVER architectural framework, the testbed offers varying notification load [26]. Defining different instants of times for the occurrence of particular event sets (a.k.a., timestamps), allows a controllable workload approach to obtain variable density and insights [27]. To achieve this, the Image Provider application is set to handle different timestamps that a follow systematic distribution along the course of the experiment time (i.e., ten minutes). As a result, the Image Provider application fetches images in the Image Data Base driven by the defined timestamp. The primary goal of this approach is to force an intended crime object detection workload to be taken by the In-Vehicle FISVER STS Fog system. This allows achieving notifications load in an assignable way.

Before defining the Image Provider application workload, it was required to identify the maximum rate of notifications that the mobile application can handle. For this, an experiment in the Typical Deployment testbed configuration was carried out. Then, the vehicle-side system increases the number of crime notification events delivered over time until the mobile application freezes by overload [28]. It was found that the Event-Driven Mobile Application is not able to handle more than 3,600 analytics in the Typical Deployment testbed. Each testbed configuration adopts four experiment sets, each one featuring different loads in offered notifications, which are calculated as a function of the maximum amount handled by the Typical Deployment testbed (3,600), namely, 30%, 60%, 90%, and 120%. Therefore, the Image Provider application is set to deliver, at different timestamps distributed along the experimental time (i.e., ten minutes), 1,080 notifications to achieve an offered notification load of 30%, 2,160 notifications for 60%, 3,240 notifications for 90%, and finally, 4,320 notifications for 120%.

B. RESULT ANALYSIS

The benchmarking focuses on obtaining performance insights in the Event-Driven Mobile Application, assuming an STS use case running on top of the Fog-FISVER architecture. Hence, efficiency savings in computation, network resource consumption, and energy resources at the Event-Driven Mobile Application side for the two (2) experiment sets are used as key performance metrics. Runtime statistics are collected by running the applications using the four workloads corresponding to each experiment. Each experiment was repeated ten (10) times. Average results are plotted on graphs, using confidence intervals of 95%.

C. CPU SAVING EFFICIENCY ANALYTICS

CPU saving efficiency stands for the capacity that a mobile device shows during its useful workload in percentage of the central processing unit residual consumption. The CPU saving efficiency measure has a direct impact on the user perception as well as on the overall operational cost caused by the system. The analysis of CPU load in the mobile device is used for benchmarking the computing efficiency. The CPU saving efficiency behavior obtained in both experiments are shown in Figure 6.

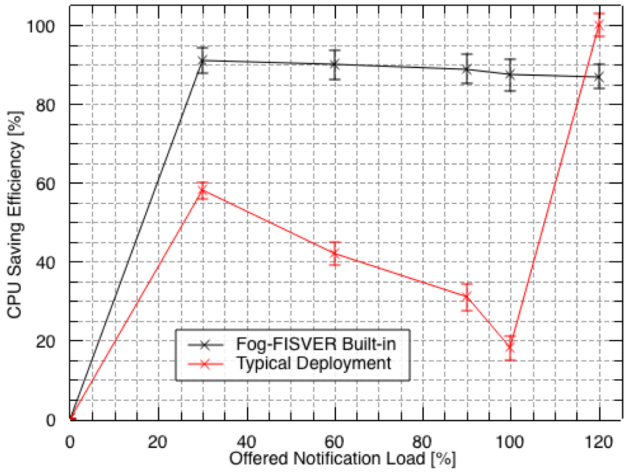

FIGURE 6. CPU Saving efficiency taken at Fog-FISVER Built- in and Typical Deployment testbed configurations.

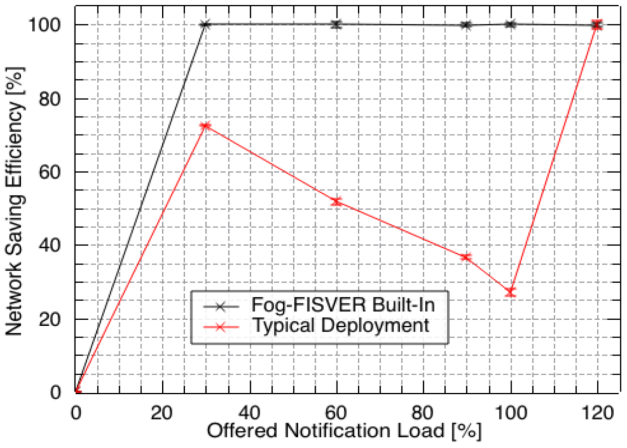

FIGURE 7. Network saving efficiency at Fog-FISVER Built-in and Typical Deployment testbed configurations.

The results shown in Figure 6 confirm that the CPU Saving efficiency in the Typical Deployment is affected exponentially with the increasing number of offered notifications. That is, the more target notifications are produced, the more crime event reports are generated and sent to the event-driven mobile application for second-level image processing task and event presentation. The average in CPU Saving efficiency in the Fog-FISVER Built-in experiments is about 88.76%, whereas the Typical Deployment experiment can only generate an average efficiency of 64.12%. Thus, the results reveal that Fog-FISVER allows improving the CPU saving efficiency by around 27.76%.

The CPU saving efficiency results suggest that the Fog-FISVER architectural framework assistance, achieved through orchestrating image analytics between both in-Vehicle FISVER STS Fog and FISVER STS Fog interworking subsystems, is beneficial for the design of smart-surveillance based STS use cases. The computational efficiency impact of the Fog-FISVER architectural framework potentially improves both cost-efficiency and agility by leaving the event-driven mobile application lightweight (i.e., only to report the police agent). It is important to emphasize the system availability impact raised by the Fog-FISVER Built-in experiments, for the reason that the Typical Deployment testbed setup is unable to reach the end of the test when the system is fed with 120% (4,320 notifications) of the offered notification load, which leads to system freezing when handling 3,600 notifications.

D. NETWORK RESOURCE SAVING EFFICIENCY ANALYTICS

Network resource saving efficiency refers to the capacity that the mobile device affords in consuming network bandwidth during its useful workload. This set of analysis plays a key role in estimating potentials of system scalability, since networking imposes challenges and issues concerning users' quality perception, system responsiveness by latencies and survivability, especially when using resource-constrained mobile devices at mission-critical use cases. With this goal in mind, the percentage of residual bandwidth (for downstream and upstream links) was measured that Fog-FISVER brings in the Typical Deployment testbed configuration over the experimental time at the mobile device, shown in Figure 7.

Figure 7 reveals that networking behavior taken by the event-driven mobile application at the Fog-FISVER Built-in set of experiments allow averaging a residual bandwidth of 99.94% (0.48 kbps) along the course of the experiment. In contrast, the Typical Deployment set of experiments allows an average of 47.99% (353.57 kbps) for residual bandwidth. Moreover, these results reveal exponentially increasing behavior by the Typical Deployment design approach, in which the mobile App is in charge of the image analytics procedures and user notification (orchestrated among Fog nodes in the Fog-FISVER approach. with lightweight behavior). The network-resource saving efficiency in Fog-FISVER Built-in set of experiments significantly outperforms the Typical Deployment results by around 51.98%, which means that the Fog-FISVER approach requires much less network bandwidth resources compared to the Typical Deployment scenario.

Therefore, the results of Figure 7 suggest that the networking optimizations of the Fog-FISVER architectural framework is a cost-beneficial way to achieve more scalable smart-surveillance based STS use cases.

E. ENERGY SAVING EFFICIENCY ANALYTICS

Device survivability is an essential aspect for reliable services, especially for mission-critical use cases, which demand nonstop service provisioning. Service disruption by mission critical factors will severely impact the target environment. In this context, the effects of security threats at the transportation service can potentially cause social turmoil, accidents, and even deaths, which must be avoided at any cost. Mobile computing reveals a trade-off by its capabilities in affording ubiquitous Internet access through the cost to the severe

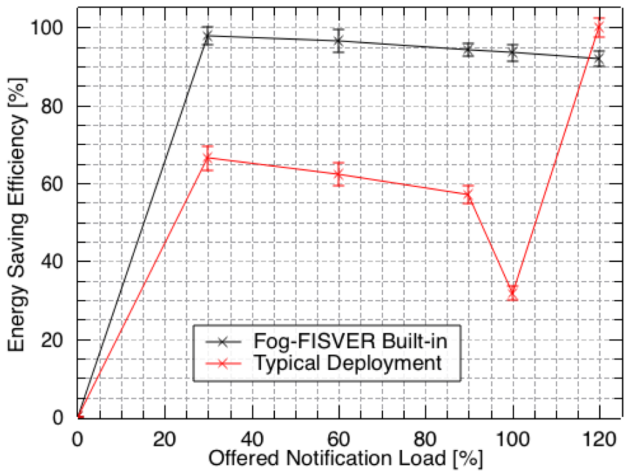

FIGURE 8. Average energy saving efficiency in Fog-FISVER Built-in and Typical Deployment testbed configurations.

energy consumption that mobile devices bring to keep service continuity via wireless connectivity over time. The energy saving efficiency reveals the ability that a mobile device affords in keeping available and accessible during processing tasks. For this set of analysis, percentage in residual battery charge is measured in both testbed configurations over the experimental time, with the goal to estimate the energy saving efficiency impact of the Fog-FISVER approach over the Typical Deployment set of experiments. The obtained results are depicted in Figure 8.

As the results shown in Fig. 8, the energy saving efficiency in the Typical Deployment set of experiments follows the same behavior as for both CPU (Figure 6) and network saving (Figure 7) efficiencies in respect to exponential projection with the increasing load of event notification. This behavior follows the same explanations as for previous analysis, because the mobile application deploys heavy image processing tasks at the Typical Deployment set of experiments, whereas the Fog-FISVER Built-in testbed applies a lightweight event-driven approach. The numerical results reveal that the residual battery charge is about 94.77% in the Fog-FISVER Built-in testbed configuration, whereas the Typical Deployment allows approximately 35.88%. Therefore, the outcomes confirm that the Fog-FISVER Built-in enhances device survivability impact over the Typical Deployment, which allows achieving an energy saving efficiency of around 62.14%.

The results highlighted through the laboratory testbed assessments, allow to consequently demonstrate the suitability and feasibility of designing smart-surveillance based STS use cases complying with the Fog-FISVER architectural framework. This conclusion is raised from the outstanding performance of the Fog-FISVER Built-in testbed setup over the Typical Deployment experiment sets, revealing enhanced perspectives regarding computational efficiency, networking cost-efficiency, system and device survivability, and scalability.

V. CONCLUSIONS AND FUTURE WORK

The main contribution of this work is the design, deployment, and performance evaluation of a fog-assistant architectural framework to afford designing smart-surveillance based STS use cases for cost-efficient crime detection in smart transportation systems. The system targets at in-Vehicle STS Fog and STS Fog Infrastructures for enhancing STS application performance. Fog-FISVER embodies a modular architecture of components that provide smart value-added procedures accessible through open Web-based interfaces. The suitability and feasibility of Fog-FISVER is assessed through laboratory testbed prototyping under public bus transportation STS use cases. The laboratory testbed prototyping outcomes indicate notable performance improvement in CPU (27.76%), network (51.98%), and energy saving (62.14%) efficiency compared to the behavior of the Typical Deployment set of experiments. The results reveal that the Smart Surveillance based STS use case, which complies with the Fog-FISVER conceptual framework, overcomes all limitations raised in Section II, through outstanding system performance and device survivability actions over the Typical Deployment. Thus, the effect of the Fog-FISVER architectural framework deployment in smart surveillance based STS use cases enforces its essential contribution in mission-critical situations by the efficiency impact in computational, networking cost optimizations, system/device survivability, and scalability.

Future work includes studying the impact of using the Fog-FISVER framework use in the resulting agility between crime threat detection till notifying the mobile application. Moreover, we aim at analyzing the performance of the Fog-FISVER Fog Infrastructures when facing the over-saturation traffic conditions. Last, but not least, it is foreseen that the deployment of the Fog-FISVER approach into a real smart surveillance based STS system, for assessments in different perspectives.

ACKNOWLEDGMENT

The experiments were carried out at NPITI/UPLab of IMD/UFRN.

REFERENCES

[1] M. A. Hossain, "Framework for a cloud-based multimedia surveillance system," *Int. J. Distrib. Sens. Netw.*, vol. 10, no. 5, p. 135257, May 2014.
[2] D. Schonfeld, "The evolution of signal processing," *IEEE Signal Process. Mag.*, vol. 27, no. 5, pp. 2–6, Sep. 2010.
[3] Homeland Security News Wire. (Sep. 23, 2011). *Study Shows Surveillance Cameras Reduce Crime, in Some Cases*. [Online]. Available: http://www.homelandsecuritynewswire.com/study-shows-surveillance-cameras-reduce-crime-some-cases
[4] G. Karagiannis *et al.*, "Vehicular networking: A survey and tutorial on requirements, architectures, challenges, standards and solutions," *IEEE Commun. Surveys Tuts.*, vol. 13, no. 4, pp. 584–616, Nov. 2011.
[5] F. Al-Doghman, Z. Chaczko, A. R. Ajayan, and R. Klempous, "A review on Fog Computing technology," in *Proc. IEEE Int. Conf. Syst., Man, Cybern. (SMC)*, Oct. 2016, pp. 1525–1530.
[6] W. Chen and T. Zhang, "Fog computing," *IEEE Internet Comput.*, vol. 21, no. 2, pp. 4–6, Mar. 2017.

[7] H. Camboim, "Cloud computing assisted smart surveillance based safe transportation system to improve crime assistance on smart cities," M.S. thesis, Syst. Comput. Graduate Program, Federal Univ. Rio Grande do Norte, Natal-RN, Brazil, 2015.
[8] Y. Wen, X. Yang, and Y. Xu, "Cloud-computing-based framework for multi-camera topology inference in smart city sensing system," in *Proc. ACM Multimedia Workshop Mobile Cloud Media Comput. (MCMC)*, New York, NY, USA, 2010, pp. 65–70. [Online]. Available: http://doi.acm.org/10.1145/1877953.187797
[9] Y.-T. Cheng, H.-A. Lin, and Y.-J. Yeh, "A flexible architecture of real-time audio transmission to heterogeneous devices for surveillance system," in *Proc. 16th Asia–Pacific Netw. Oper. Manage. Symp. (APNOMS)*, Sep. 2014, pp. 1–4.
[10] A. K. Paul and J. S. Park, "Multiclass object recognition using smart phone and cloud computing for augmented reality and video surveillance applications," in *Proc. Int. Conf. Inform., Electron. Vis. (ICIEV)*, May 2013, pp. 1–6.
[11] Y. Jian, W. Xin, Z. Xue, and D. Z. You, "Cloud computing and visual attention based object detection for power substation surveillance robots," in *Proc. IEEE 28th Can. Conf. Elect. Comput. Eng. (CCECE)*, May 2015, pp. 337–342.
[12] D. Rodriguez-Silva, L. Adkinson-Orellana, and F. Gonz'lez-Castano, I. Armino-Franco, and D. Gonz'lez-Martinez, "Video surveillance based on cloud storage," in *Proc. IEEE 5th Int. Conf. Cloud Comput. (CLOUD)*, Jun. 2012, pp. 991–992.
[13] F. Zhu, H. Qiu, and Z. Y. Song, "Rhizome: A middle-ware for cloud vision computing framework," in *Proc. IET Int. Conf. Smart Sustain. City (ICSSC)*, Shanghai, China, 2013, pp. 193–197, doi: 10.1049/cp.2013.2023.
[14] X. Chen, J.-B. Xu, and W.-Q. Guo, "The research about video surveillance platform based on cloud computing," in *Proc. Int. Conf. Mach. Learn. (ICMLC)*, vol. 2. Jul. 2013, pp. 979–983.
[15] N. Chen, Y. Chen, Y. You, H. Ling, P. Liang, and R. Zimmermann, "Dynamic urban surveillance video stream processing using fog computing," in *Proc. IEEE 2nd Int. Conf. Multimedia Big Data (BigMM)*, Apr. 2016, pp. 105–112.
[16] N. Chen, Y. Chen, X. Ye, H. Ling, S. Song, and C. T. Huang, "Smart city surveillance in fog computing," in *Advances in Mobile Cloud Computing and Big Data*, vol. 22, C. Mavromoustakis, G. Mastorakis, and C. Dobre, Eds. Cham, Switzerland: Springer, 2017, pp. 203–226. [Online]. Available: http://dx.doi.org/10.1007/978-3-319-45145-99
[17] V. Lanaria. *Netatmo Presence Smart Surveillance Camera Can Tell if it's an Intruder or Your Cat Lurking Outside Your Home*. Accessed: Jan. 8, 2016. [Online]. Available: http://www.techtimes.com/articles/122385/20160108/netatmo-presence-smart-surveillance-camera-can-tell-if-it-s-an-intruder-or-your-cat-lurking-outside-your-home.htm
[18] Skywatch Inc. (2015). *Announces Smart Video Surveillance Cloud Service at Cloud Expo*. Accessed: Jun. 15, 2015. [Online]. Available: http://www.marketwired.com/press-release/skywatch-inc-announces-smart-video-surveillance-cloud-service-at-cloud-expo-2015-2029404.htm
[19] *BrickCom IVA Technology*. Accessed: Dec. 12, 2016. [Online]. Available: http://www.brickcom.com/products/DetailView.php?series=Other_Applications&modelname=IVA#product-features
[20] *Smartvue*. Accessed: Dec. 12, 2016. [Online]. Available: http://smartvue.com
[21] R. Brunelli, *Template Matching Techniques in Computer Vision: Theory and Practice*. Hoboken, NJ, USA: Wiley, 2009.
[22] T. Winkler and B. Rinner, "Pervasive smart camera networks exploiting heterogeneous wireless channels," in *Proc. IEEE Int. Conf. Pervasive Comput. Commun. (PerCom)*, Mar. 2009, pp. 1–4.
[23] D. D. Hall and J. Budinger, "Nextgen ats communications, navigation, and surveillance test bed," in *Proc. IEEE/AIAA 26th Digit. Avion. Syst. Conf.*, Oct. 2007, pp. 4.A.1-1–4.A.1-11.
[24] T. Arulogun, A. Adigun, O. Okediran, and R. Ganiyu, "Design and development of a security surveillance system based on wireless sensor network," *Int. J. Innov. Sci., Eng. Technol.*, vol. 1, no. 4, pp. 283–291, Jun. 2014.
[25] T. M. Inc. *Train a Cascade Object Detector*. Accessed: Nov. 2014. [Online]. Available: http://www.mathworks.se/help/vision/ug/train-a-cascade-object-detector.html#btugex8
[26] G. Caterina, I. Hunter, and J. Soraghan, "An embedded smart surveillance system for target tracking using a PTZ camera," in *Proc. 4th Eur. Edu. Res. Conf. (EDERC)*, Dec. 2010, pp. 165–169.
[27] A. Hampapur, L. Brown, J. Connell, S. Pankanti, A. Senior, and Y. Tian, "Smart surveillance: Applications, technologies and implications," in *Proc. 4th Pacific Rim Conf. Multimedia, Inf., Commun. Signal Process.*, vol. 2. Dec. 2003, pp. 1133–1138.
[28] V. Onut, D. Aldridge, M. Mindel, and S. Perelgut, "Smart surveillance system applications," in *Proc. Conf. Center Adv. Stud. Collaborative Res. (CASCON)*, Riverton, NJ, USA, 2010, pp. 430–432. [Online]. Available: http://dx.doi.org/10.1145/1923947.1924031

AUGUSTO J. V. NETO received the Ph.D. degree in informatics engineering from the University of Coimbra, Portugal, in 2008. He is currently an Associate Professor with the Department of Informatics and Applied Mathematics, Federal University of Rio Grande do Norte, Natal, Brazil, and a member of the Instituto de Telecomunicações, Portugal. He has authored and co-authored many papers published in national/international conference proceedings and journals in the field of computer networks and telecommunications. He coordinates and participates in research projects funded by national and international agencies, mainly with future Internet, 5G networks, mobile computing, smart spaces, SDN, NFV, IoT, and cloud/edge computing. He is involved in the organization of several international conferences and workshops, as well as serving as the Guest Editor for special issues of peer-reviewed scholarly journals.

ZHONGLIANG ZHAO received the Ph.D. degree from the University of Bern in 2014. He was a Senior Researcher with the University of Bern. He has been active as multiple work package leaders in the EU FP7 project Mobile Cloud Networking, the Co-PI of the Sino-Swiss Science and Technology Cooperation Project M3WSN. He is currently the Technical Coordinator of the Swiss National Science Foundation Project SwissSenseSynergy and industry projects. His research interests include IoT, mobile cloud/edge/fog computing, SDN/NFV, urban computing, machine learning, and indoor localization.

JOEL J. P. C. RODRIGUES (S'01–M'06–SM'06) has been a Professor with the University of Beira Interior, Portugal, and a Visiting Professor with the University of Fortaleza, Brazil. is currently a Professor and a Senior Researcher with the National Institute of Telecommunications (Inatel), Brazil, and a Senior Researcher with the Instituto de Telecomunicações, Portugal. He has authored or co-authored over 550 papers in refereed international journals and conferences, three books, and two patents. He is the Leader of the Internet of Things Research Group (CNPq), a member of the IEEE ComSoc Board of Governors as the Director for Conference Development, the IEEE ComSoc Distinguished Lecturer, the President of the Scientific Council at ParkUrbis–Covilhã Science and Technology Park, the Past-Chair of the IEEE ComSoc Technical Committee on e-Health, the Past-Chair of the IEEE ComSoc Technical Committee on Communications Software, Steering Committee Member of the IEEE Life Sciences Technical Community and Publications Co-Chair, and a Member Representative of the IEEE Communications Society on the IEEE Biometrics Council. He is a Licensed Professional Engineer (as a Senior Member), a member of the Internet Society, and a Senior Member ACM. He is a member of many international TPCs and participated in several international conferences organization. He had been awarded several Outstanding Leadership and Outstanding Service Awards by the IEEE Communications Society and several best papers awards. He has been the General Chair and the TPC Chair of many international conferences, including the IEEE ICC, the GLOBECOM, and HEALTHCOM. He is the Editor-In-Chief of three international journals and an Editorial Board Member of several high-reputed journals.

HUGO BARROS CAMBOIM received the B.Sc. degree from the State University of Paraíba, Brazil, in 2010, the M.Sc. degree from the Federal University of Rio Grande do Norte, Brazil, in 2016, under the supervision of Dr. A. Neto, Brazil. He was a Lecturer with the State University of Paraíba, cooperating in research projects in the field of software development for education. His research interests include distributed systems, ubiquitous computing, and smart transport safety.

TORSTEN BRAUN received the master's and Ph.D. degrees from the University of Karlsruhe, Germany, in 1990 and 1993, respectively. From 1994 to 1995, he was a Guest Scientist with INRIA Sophia Antipolis, France. From 1995 to 1997, he was a Project Leader and a Senior Consultant with the IBM European Networking Center, Heidelberg, Germany. Since 1998, he has been a Full Professor of computer science with the Institute of Computer Science, University of Bern, Switzerland, heading the Communications and Distributed Systems Research Group. He has been the Deputy Dean of the Faculty of Science with the University of Bern, since 2017. He has been a Board Member of Swiss Education and Research Network (SWITCH) since 2000 and the Vice President of the SWITCH Foundation since 2011.