IoT

Review

A Survey of Security Architectures for Edge Computing-Based IoT

Elahe Fazeldehkordi * and Tor-Morten Grønli

Mobile Technology Lab (MOTEL), Department of Technology, Kristiania University College, Kirkegata 24-26, 0153 Oslo, Norway; Tor-Morten.Gronli@kristiania.no
* Correspondence: Elahe.Fazeldehkordi@kristiania.no

Abstract: The Internet of Things (IoT) is an innovative scheme providing massive applications that have become part of our daily lives. The number of IoT and connected devices are growing rapidly. However, transferring the corresponding huge, generated data from these IoT devices to the cloud produces challenges in terms of latency, bandwidth and network resources, data transmission costs, long transmission times leading to higher power consumption of IoT devices, service availability, as well as security and privacy issues. Edge computing (EC) is a promising strategy to overcome these challenges by bringing data processing and storage close to end users and IoT devices. In this paper, we first provide a comprehensive definition of edge computing and similar computing paradigms, including their similarities and differences. Then, we extensively discuss the major security and privacy attacks and threats in the context of EC-based IoT and provide possible countermeasures and solutions. Next, we propose a secure EC-based architecture for IoT applications. Furthermore, an application scenario of edge computing in IoT is introduced, and the advantages/disadvantages of the scenario based on edge computing and cloud computing are discussed. Finally, we discuss the most prominent security and privacy issues that can occur in EC-based IoT scenarios.

Keywords: edge computing (EC); EC-based IoT; Internet of Things (IoT); security; privacy; architecture

Citation: Fazeldehkordi, E.; Grønli, T.-M. A Survey of Security Architectures for Edge Computing-Based IoT. *IoT* **2022**, *3*, 332–365. https://doi.org/10.3390/iot3030019

Academic Editor: Javier Berrocal

Received: 16 February 2022
Accepted: 21 June 2022
Published: 30 June 2022

1. Introduction

Over the past decades, cloud computing [1,2] has been largely developed and applied due to its high cost-efficiency and flexibility achieved through a combination of computing, storage, and network management functions that work in a centralized manner. Existing centralized cloud computing architecture is facing severe challenges owing to the fast development of the Internet of Things (IoT) and mobile Internet applications. For obtaining sophisticated applications, mobile devices connected to remote centralized cloud servers impose additional loads on both radio access networks (RANs) and backhaul networks that result in high latency [3]. Furthermore, with the progressing development of information technology, IoT is playing an important role in all aspects of current modern life [4]. With the advent of IoT, new challenges have arisen, such as stringent latency, capacity constraints, resource-constrained devices, uninterrupted services with intermittent connectivity, and enhanced security, which centralized cloud computing architecture cannot adequately satisfy [5].

IoT refers to the modern interaction and communication network infrastructure of millions of physical objects (i.e., things) embedded with interconnected sensors/devices that can produce, collect, and exchange different data amongst themselves [6–10]. Therefore, more and more sensors and devices are being interconnected through IoT technology, and these sensors and devices will generate enormous amounts of data and require further processing and analytics, providing intelligence to both service providers and users. In conventional cloud computing, all data must be uploaded to centralized servers, and after computation, the results should be sent back to the sensors and devices. This process will require prohibitively high network bandwidth and the data transmission costs of bandwidth and resources.

In addition, by increasing the size of the data, the network performance will deteriorate. Furthermore, since in a conventional cloud computing-based service, the computation processes need to be uploaded to the cloud servers that are already far from the end users, and the limited bandwidth and network resources are occupied by massive data transmissions, there will be large latency within the networks. In time-sensitive IoT applications, meaning that they require very short response times, such as smart transportation [11], smart electricity grid [12,13], smart city [14–16], etc., large latency is more critical, and it is unacceptable. This is an important issue for IoT since these applications will have an impact on safety and emergency response.

Moreover, most IoT devices (smartphones, wearable watches, etc.) still have fundamental challenges such as limited memory size and limited battery life. Balancing power consumption by transferring the energy-consuming computation tasks of applications to devices that have higher power and computational capabilities is necessary to be able to extend the battery life of devices. In addition, processing data in computation nodes with a short distance to the end user will reduce transmission time, leading to a reduction in power consumption. In contrast, in cloud computing-based services, network traffic affects the data transmission speed, and heavy traffic results in long transmission times, therefore increasing power consumption costs. Hence, offloading the energy-consuming computation of applications to the edge of networks [17] is a critical issue that should be considered in order to extend the battery life of IoT devices. Recent research has investigated the effective exploitation of capabilities at the edge of networks in order to support IoT and its requirements [9].

In edge computing (EC), computing and storage of massive data produced by different types of IoT devices can be performed at the network edge close to the user instead of transferring them to a centralized cloud infrastructure [18–26]. Since the locations of edge computing nodes are close to end users, the peak in traffic flows will be reduced. In addition, the bandwidth requirements of the centralized network and the transmission latency during data computing or storage in IoT will be significantly mitigated. Distributing computation nodes deployed at the edge allow offloading of traffic and computational pressure from the centralized cloud, and it provides faster response times for IoT applications and greater quality services in comparison with cloud computing. Reducing the overall delay of the system and the demand for communication bandwidth is very efficient and would improve the overall performance of the system.

Transferring the data directly to the cloud platform requires substantial operational costs for data migration, good bandwidth, and delay characteristics. By reducing data uploading volume in EC-based IoT, data migration volume, bandwidth consumption, and latency will be reduced; therefore, operational costs will be decreased accordingly.

Another advantage of edge computing is protecting data security and privacy [27–30]. Even though cloud platform service providers offer a comprehensive system of centralized data security protection solutions, if centralized stored data becomes leaked, it will have serious consequences. Edge computing in IoT allows deploying the most appropriate security solutions in the local vicinity, so it reduces the risk of data leakage during transmission, as well as the data volume stored in the cloud platform; therefore, security and privacy risks will be minimized.

Furthermore, in edge computing, the computational and communication overhead from IoT devices with limited battery or power supply will be migrated to edge nodes with significant power resources. Therefore, the lifetime of the IoT devices with a limited battery will be extended, resulting in an increased lifetime of the entire IoT network. Thus, edge computing is more suitable for IoT services; it can offer secure and efficient services for a large number of end users, and architectures based on edge computing can be considered for future IoT infrastructure [10].

There are some existing related studies in the field of edge computing that have provided a survey of papers and architectures in the area of edge computing, fog computing, or MEC. The authors in [31] presented a comprehensive survey on fog computing and related computing models and their differences and similarities. They also provide a summary and taxonomy of research into fog computing and its related computing models. Some works have focused on security and privacy issues and related countermeasures in edge computing paradigms [32,33]. Other works have proposed architectures for EC-based IoT [34,35]. Inspired by these works and using other materials from top journals/conferences, high-cited works, and references in the field, our study provides a detailed and up-to-date analysis of several subjects from a holistic perspective.

The main contributions of this article are summarized as follows:

1. We provide an overview and definition of edge computing and its relationship/difference with/from other similar computing models, such as fog computing, cloud computing, cloudlets, and MEC.
2. We present attacks and threats of EC-based IoT. Then, we discuss the possible solutions and countermeasures at different network layers and for different security and privacy issues.
3. We propose a secure edge computing-based architecture for IoT infrastructure through many research achievements concerning edge computing in IoT and define an application of edge computing in IoT for the architecture.

The remainder of this paper is structured as follows: in Section 2, we discuss cloud computing and other similar computing paradigms and compare edge computing with other similar computing paradigms. Section 3 provides classifications of security and privacy attacks and threats for EC-based IoT. Section 4 describes possible security solutions and countermeasures. It also provides a comprehensive analysis of security and privacy issues for EC-based IoT. Section 5 presents an edge computing-based architecture for IoT infrastructure. Section 6 defines IoT scenarios based on cloud and edge computing as an application of the proposed architecture. Section 7 discusses the advantages/disadvantages of scenarios. Section 8 highlights the main challenges. Finally, Section 9 concludes this article.

2. Edge Computing and Related Computing Paradigms

This section focuses on edge computing, types of edge computing technology, and related computing paradigms. We compare edge computing with related computing paradigms and also provide a comparison of different types of edge computing technology.

2.1. Cloud Computing

Using cloud computing, the access and abilities of computing, storage, and network infrastructure have been developed for applications [31]. According to the definition of the national institute of standards and technology (NIST), cloud computing is a pattern to further omnipresent and on-demand network access to shared resources [1]. Cloud data centers (provided by cloud providers, such as Google, IBM, Microsoft, and Amazon) offer virtualized resources that are highly accessible, scalable, and can be reconfigured dynamically; this reconfigurability helps cloud computing to offer services with a pay-as-you-go cost model [36]. With the pay-as-you-go cost model, users are able to conveniently access remote computing resources and data management services, and they will only be charged for the number of resources that they use. Cloud offers services such as infrastructure as a service (IaaS), platform as a service (PaaS), and software as a service (SaaS) [31]. A variety of these services can be used by application developers based on their requirements for the applications that they develop.

The initial aim of cloud computing was to allow users to have access to computing resources for omnipresent computing. Although cloud computing has assisted in achieving this aim, it may take a long time to access cloud-based applications, which is not feasible for some critical applications that require very low latency. The growing number of connected devices and the amount of data generated at the network edge require cloud resources to be in close proximity to the data generated place. Due to the high demand for high-bandwidth, low-latency, geographically distributed, and privacy-sensitive data, there is an essential need for computing models that can take place in close proximity to connected devices to be able to support the mentioned demands. For this purpose, edge computing has been proposed [37,38]. In the following, we explain various computing paradigms and compare them.

2.2. Mobile Computing

The development of mobile computing (MC) has influenced the advancement of cloud computing. In mobile computing (also called nomadic computing), computing is accomplished through mobile and portable devices [31]. One of the applications of mobile computing is pervasive context-aware applications, such as location-based reminders.

Mobile computing has the advantage of distributed computing architecture. With this architecture, mobile machines are able to operate in decentralized locations. Nevertheless, mobile computing has many disadvantages such as poor resource constraints, communication latency, the balance between autonomy and interdependence (common in all distributed architectures), and mobile clients' need to efficiently adapt to changing environments [39]. Because of these drawbacks, mobile computing is often not suitable for applications with low latency or robustness requirements or applications that generate, process, and store large amounts of data on devices.

The scale and scope of mobile computing have been developed by fog (see Section 2.4.3) and cloud computing. Computation in fog and cloud computing is not limited to a local network. Mobile computing only requires the hardware of mobile devices. Fog and cloud computing, however, need more rich resource hardware along with virtualization abilities. Security in mobile computing should be given via securing the mobile device. Mobile computing is more resource-constrained compared with fog and cloud computing; however, recently, advancements in wireless protocols and mobile hardware have notably lessened this gap.

2.3. Mobile Cloud Computing

The combination of mobile computing and cloud computing is called mobile cloud computing (MCC), in which both data processing and data storage happen in the cloud outside the mobile device [31,40]. The focus of MCC is on the relationship between cloud service providers and cloud service users of mobile devices.

In MCC, resource-constrained mobile devices are able to use the rich resources of cloud services. Most part of the computation in MCC has shifted from mobile devices to the cloud. MCC is able to run applications with intensive computations and increases the mobile device's battery life. Specifications and abilities of both mobile and cloud computing have been used in MCC. By applying a combination of mobile and cloud computing, computing resources are highly available in MCC compared with mobile computing, which is resource constrained. This is beneficial for applications that require high computation, e.g., mobile augmented reality. Cloud-based services in MCC are considerably more available compared with mobile computing. However, mobile devices can function the computation in MCC, but for operating high-computation services, MCC relies on cloud services. In MCC, security must be provided on both mobile devices and the cloud.

MCC also has the same restrictions as mobile and cloud computing. MCC has a centralized architecture that might not be suited to applications with the desire for the pervasiveness of devices. Moreover, accessing cloud-based services in both cloud computing and MCC uses a WAN connection; therefore, applications running on these platforms need to be always connected to the Internet, which introduces connectivity challenges. In addition, offloading computation to the cloud results in high latency, which, as we discussed, is not suitable for delay-sensitive applications.

2.4. Edge Computing

The storage, management, and processing power of data generated by connected devices have been enhanced by edge computing. In contrast to MCC, edge computing has been placed in close proximity to IoT devices at the edge of the network. It is worth mentioning that edge is as close as one hop from IoT devices and not placed on those devices. Nevertheless, it is possible that edge is more than one hop far from IoT devices in the local IoT network. According to OpenEdge Computing, computation in edge computing is performed at the edge of the network via small data centers in the vicinity of users [41]. The main purpose of edge computing is to provide storage and computation resources in close proximity to the user in a way that is omnipresent [41]. Edge computing is a pivotal computing model for IoT devices—the filtering, preprocessing, and aggregating of IoT data is performed using cloud centers placed in the vicinity of IoT devices [42]. The main advantages of edge computing are as follows [31,34,42,43]:

- Improve system performance: the most important advantage of edge computing in IoT is achieving ms-level of data processing. Edge computing reduces the overall delay of the system and the demand for communication bandwidth and improves the overall performance of the system.
- Protect data security and privacy: cloud platform service providers offer a comprehensive system of centralized data security protection solutions to their customers. However, once centralized stored data becomes leaked, it will lead to serious consequences. In contrast, edge computing allows deploying the most appropriate security solutions in the local vicinity, and most of the computation can be performed on the edge of the network, so less data need to be transferred. Therefore, it reduces the risk of data leakage during the transmission and the amount of stored data in the cloud platform; thus, security and privacy risks will be decreased. IoT devices collect a large amount of data that can contain some sensitive information (e.g., global positioning system (GPS) data, streams from cameras/microphones). An application might need this information to run complex analytics in the cloud; it is important to preserve the privacy of sensitive context once the data leaves where it was generated. Using edge computing, the sensitive data can be preprocessed onsite and passed through a first layer of anonymization, and then the privacy complaint data can be sent to the cloud for further analysis.
- Reduce latency: latency in edge computing has been reduced compared with MCC and cloud computing because of the proximity to users. However, if the local computation unit is not powerful enough, the latency in edge computing can be worse than in MCC and cloud computing.
- Reduce operational costs: transferring data directly to the cloud platform requires substantial operational costs for data transmission, good bandwidth, and delay characteristics. In contrast, edge computing can reduce data uploading volume; therefore, data transmission volume, bandwidth consumption, and latency will be reduced, consequently reducing operational costs.
- High service availability: availability of services is also superior in edge computing; there is no waiting time for a centralized platform to provide the services, and there is no limitation of resources, such as in traditional mobile computing. In contrast to MACC (see Section 2.4.2), edge computing contains small data centers, while MACC basically does not require a data center. Therefore, the availability of services is higher in edge computing. In addition, edge computing can form peer-to-peer and cloud computing hybrid models, so it benefits from broader computing capabilities than does MACC [44].
- Be robust to connectivity issues: when part of the computations can be run directly on the edge, applications will not be disrupted by limited or intermittent network connectivity. This is especially beneficial when applications are running on remote locations with poor network coverage. It can also reduce expensive costs related to connectivity technologies, such as cellular technologies.

Where is Edge?

In the telecommunications industry, the term edge generally alludes to 4G/5G base stations, radio access networks (RANs), and Internet service provider (ISP) access/edge networks. However, in the IoT landscape [42,45], the term edge refers to the local network where sensors and IoT devices are placed.

The edge refers to the immediate first hop from the IoT devices but not the IoT nodes themselves, e.g., the Wi-Fi access points or gateways. The computing paradigm that the computation is performed on IoT devices is called mist computing (see Section 2.4.5). In this paper, we consider mist computing as one of the technologies of edge computing.

2.4.1. Cloudlet Computing

The first edge computing concept bringing computation/storage closer to user equipment (UE)s was proposed in 2009 by Carnegie Mellon University [43,46]. Cloudlet is sometimes referred to as micro data center (MDC) in some studies [47]. The idea behind the cloudlet is placing a computer or a cluster of computers with trusted high resources and computation power and a strong connection to the Internet at strategic locations close to edge devices to provide both computation resources and storage for the UEs in the vicinity [46]. Cloudlets are small-scale data centers (miniature clouds) that are usually located one hop from mobile devices. The purpose of cloudlet computing is offloading computation from mobile devices to virtual machine (VM)-based cloudlets placed on the edge of the network and in the local area [48].

Cloudlet is a small cloud in close proximity to mobile devices. Cloud service providers who have a plan to offer accessible services in the proximity of mobile devices can be operators of cloudlet computing. Cloudlets can be enabled by network infrastructure owners, such as AT&T, Nokia, etc., with virtualization capacity placed in close proximity to mobile devices, with small-scale hardware sizes in contrast to huge data centers in cloud computing. Computing resources in cloudlets are more moderate because of the small sizes of cloudlets, but latency and energy consumption are lower compared with cloud computing.

A disadvantage of cloudlets is that they are predominantly accessible through Wi-Fi connection, and mobile UEs have to switch between Wi-Fi and the mobile network to use cloudlet services [4]. In addition, cloudlets are not an inherent part of the mobile network, and Wi-Fi only has local coverage with limited mobility support, so it is hard to attain good quality of service (QoS) for mobile UEs, the same as in the case of MCC.

Mobile cloudlets are a similar concept to cloudlets. In mobile cloudlets, cloudlets are a group of close mobile devices connected wirelessly, for instance, using Wi-Fi or Bluetooth [49]; mobile devices here can be providers or clients of computing services.

2.4.2. Mobile Ad Hoc Cloud Computing

The other option to enable cloud computing at the edge is doing computing directly at the UEs through an ad hoc cloud that allows several UEs in proximity to combine their computation power; therefore, they can process high-demanding applications locally [43,50–59].

MCC is pervasive but is not very suitable in the absence of a centralized cloud or infrastructure. An ad hoc mobile network is an impermanent and dynamic network of nodes that has been created via routing and transport protocols. Mobile ad hoc cloud computing (MACC) is the most decentralized form of a network [60]. In MACC, mobile devices create a highly dynamic network topology; network adaptation for more devices to frequently leave/join the network is necessary. Ad hoc mobile devices are also able to create clouds to use for networking, storage, and computing. Unmanned vehicular systems and group live video streaming are examples of MACC use cases.

Compared with cloudlets, virtualization with VM capability configuration is required in cloudlets, but such infrastructure is not required in MACC. Mobility has been supported in both cloudlet computing and MACC; however, real-time IoT applications are not supported in resource-constrained MACC. Tasks in the cloudlet have been divided among cloudlet nodes placed in the vicinity of mobile devices in order to bear local services for mobile customers [61].

Resources in MACC have an ad hoc nature; therefore, MACC is basically distinct from cloud computing. Mobile devices in MACC act as data storage, data providers, and processing devices, and because of the lack of network infrastructure, they also control routing traffic amongst themselves. Local mobile resources have been pooled to create an ad hoc cloud; therefore, MACC can offer high computation. These attributes are distinct from the target users, connectivity, and architecture in cloud computing.

The service access method, hardware, and distance from users are different in MACC and MCC. The computation in MACC happens on mobile devices; however, in MCC, the computation is away from mobile devices. In MACC, only mobile devices have been required to operate; however, in MCC, extensive data centers used for cloud computing are also needed. Therefore, computation power in MCC is high, but latency is also high in MCC. Security in MACC is only achieved through mobile devices but achieving trust can be challenging in the absence of a secure collaboration framework. Services in MACC are only accessible using mobile devices; the connection between mobile devices is through Bluetooth, Wi-Fi, or other cellular protocols.

A mobile ad hoc network (MANET) is a similar concept to MACC. In MANETs, mobile host devices are connected to each other with a single hop and without base stations [62]. MANET devices create dynamic networks but not necessarily a cloud, meaning that the computation or storage resource pools are not necessary for MANETs. Solutions such as redundancy and broadcasting to MANETs are applicable to MACC.

2.4.3. Fog Computing

Another concept of edge computing is fog computing. The fog computing paradigm (shortly often abbreviated as “fog” in the literature) was introduced by Cisco in 2012 to enable the processing of applications on billions of connected devices at the edge of the network [38]. Fog computing is defined as decentralizing a computing infrastructure by extending the cloud by locating nodes between the cloud and edge devices [63,64]. This puts data, computation, storage, and applications closer to the user or IoT device where the data needs processing; therefore, a fog will be created outside the centralized cloud and reduces the necessary data transfer times to process data.

As a consequence, computation, storage, networking, decision-making, and data management take place both in the cloud and throughout the IoT-to-cloud route while data travel over to the cloud (preferably in close proximity to IoT devices). For example, in intelligent transportation systems (ITSs), GPS data can be compressed at the edge before transferring to the cloud [65]. According to the definition of the OpenFog Consortium, fog computing is [37]: “a horizontal system-level architecture that distributes computing, storage, control and networking functions closer to the users along a cloud-to-thing continuum.” Computing functions in fog computing have been distributed amongst various platforms and industries due to the “horizontal” platform [66]. To be able to fulfill the requirements of data-driven operators/users, a flexible platform has been provided in fog computing. IoT can be strongly supported by fog computing.

Compared with traditional cloud computing, fog nodes are placed in the vicinity of IoT source nodes, which causes significantly low latency. Node locations are less centralized in fog computing, as opposed to cloud data centers, which are centralized. Fog nodes are widely distributed geographically. In fog computing, security is achieved via the edge or in the fog node locations compared with cloud computing, in which security mechanisms are provided in the location of cloud data centers. Because fog computing is decentralized, devices can be served as either fog computing nodes themselves or as clients of the fog that use fog resources.

The main difference between cloud and fog computing is the scale of hardware components related to these computing paradigms. Cloud computing provides highly available computing resources with relatively high-power consumption, while fog computing provides computing resources with medium availability and lower power consumption [67]. Cloud computing usually utilizes large data centers, whereas, in fog computing, there are small servers, routers, switches, gateways, set-top boxes, or access points. Hardware in fog computing occupies much less space compared with the one in cloud computing; therefore, it can be placed closer to users. Fog computing is accessible via connected devices from the edge to the network core, while cloud computing is accessible via the network core. In addition, fog-based services can work without continuous Internet connectivity (with low/no Internet connection), meaning that the services are able to work independently, then necessary updates will be sent to the cloud anytime that there is an Internet connection. However, in cloud computing, devices need to be connected to the Internet while the cloud service is running.

In fog, computation, communication, storage, control, and decision-making have been distributed closer to IoT devices, helping devices to measure, monitor, process, analyze, and react [37]. Many industries, such as energy, manufacturing, transportation, healthcare, smart cities, etc., could benefit from fog computing.

Compared with MACC, MACC is more suitable for highly decentralized and dynamic networks in which there might be no Internet connection. In MACC, connected devices are mostly not centralized; therefore, devices form a more dynamic network [68].

Compared with cloudlets, however, cloudlet computing fits well with the mobile cloudlet–cloud framework, but fog computing can support large amounts of traffic; in addition, resources in fog can be located anywhere throughout the things-to-cloud path [69].

Fog and edge computing are not identical, even though both perform the computation and storage in the edge close to end users. The OpenFog Consortium states this distinction as the hierarchical nature of fog computing, allowing computing, networking, storage, control, and acceleration anywhere along the cloud-to-things continuum; however, in edge computing, computing is performed only at the edge [37].

The most significant drawback of the above-mentioned edge computing concepts (cloudlet, ad hoc clouds, and fog computing) is that the computing is not integrated into the architecture of a mobile network; therefore, QoS and quality of experience (QoE) for users are not completely guaranteed [43]. One concept that can integrate cloud capabilities into a mobile network is the cloud radio access network (C-RAN) [70]. The C-RAN uses the idea of distributed protocol stack [71], in which some layers of the protocol stack have changed their place from distributed radio remote heads (RRHs) to centralized base band units (BBUs) (see [72]).

A fog radio access network (Fog RAN/F-RAN) is a form of fog computing that is also able to be combined with mobile technologies; more details can be found in [73]. F-RAN and C-RAN can be both used for mobile networks with base stations; they can be implemented for 5G-related mobile technology deployments [73] and are more energy efficient for network operations. For more information about F-RAN, we encourage readers to refer to the reference [74].

2.4.4. Multi-Access Edge Computing

Another concept merging edge computing and mobile network architecture was proposed by the newly created (2014) industry specification group (ISG) within the European telecommunications standards institute (ETSI) [75]. The solution outlined by ETSI is known as mobile edge computing (MEC), which has been supported by prominent mobile operators and manufacturers, such as NTT DoCoMo, Vodafone, TELECOM Italia, IBM, Nokia, Huawei, Intel, and other companies. At the MEC World Congress 2016, MEC ISG renamed mobile edge computing to "multi-access edge computing" to reflect the growing interests of noncellular operators and to include a wider scope of applications not only limited to mobile device-specific tasks.

According to ETSI, MEC is a platform that offers application developers and content providers cloud computing capabilities and an IT service environment within RAN in 4G and 5G at the edge of the network [76].

In MEC, edge computing functionalities can be appended to existing base stations using RAN operators. In MEC, data centers are small with virtualization capacity. Compared with cloud computing, available computing resources in MEC are moderate because of the underlying hardware.

Furthermore, low-latency applications, as well as delay-sensitive critical applications [75], can be supported in MEC. A personalized and contextualized experience has been offered to mobile users using MEC applications because these applications profit from real-time radio and network information.

Connectivity in MEC has been established via WAN, Wi-Fi, or cellular connections. The focus of MEC is on RAN-based network infrastructure providers. It is anticipated that MEC benefits significantly from the 5G platform [75] since a broad range of mobile devices with lower latency and higher bandwidth can be supported by 5G. In addition to 5G technologies, MEC also benefits from using software-defined networking (SDN) and network function virtualization (NFV) capabilities [77,78].

2.4.5. Mist Computing

Recently, mist computing has been presented that describes dispersed computing at the extreme edge of connected devices (the IoT devices themselves) [79,80]. Mist computing could be considered the earliest computing hop in the cloud–fog–IoT sequence; informally, it is called "IoT computing" or "things computing." An IoT device can be a wearable watch, smartphone, smart fridge, etc. Mist computing extends computing, storage, and networking across the fog through the things. Mist computing can be considered a superset of MACC; since, in mist, the networking is not necessarily ad hoc, and the devices may not be mobile devices.

Research shows that using mist computing, the load in traditional Wi-Fi infrastructures for video dissemination applications can be reduced [81], users' data privacy can be preserved through local processing [82], or virtualized instances on single-board computers can be deployed efficiently [83].

Other than the computing paradigms explained above, cloud of things (CoT) [84–87] and edge cloud [88–91] are also other similar computing paradigms that have been mentioned in some studies.

3. Security and Privacy Attacks and Threats in EC-Based IoT

In this section, we describe the possible key security and privacy attacks, their types, and their sources at different levels and layers (EC devices, communication and EC servers/nodes, and cloud servers) of edge computing in IoT networks.

1. *Malicious hardware/software injection:* unauthorized software/hardware components to the communication or EC node levels can be added by attackers that inject malicious inputs into the EC servers. Then, adversaries will be able to exploit service providers to perform hacking processes on their behalf, such as bypassing authentication, stealing data, reporting false data, or exposing database integrity [33,92,93]. Hardware injection attacks have several classifications, and we will further investigate the most common ones.

- *Node replication*: this occurs when adversaries inject a new malicious node into an existing set of nodes by replicating one node's ID number. Then, attackers will be able to corrupt, steal, or misdirect data packets arriving at the malicious replica. The required access to extract cryptographic shared keys can be obtained by attackers causing severe damage to the system. Moreover, by implementing node revocation protocols, legitimate EC nodes can be revoked by node replicas [33,92]. This attack is considered an active attack [94–96].
- If attackers gain illegitimate access to integrated circuits (ICs), they can appear as *hardware trojan*. Attackers will be able to control the circuit and access data or even software running on these ICs. There are two types of Trojans: (1) internally activated Trojans, which can be activated by satisfying a particular condition inside the Ics; and (2) externally activated Trojans, which can be activated using sensors or antennas that interact with the outside world [33,92].
- Attackers can also *camouflage* by injecting a fake EC node into the network or attack an authorized node to be able to hide at the edge level. This counterfeit/modified EC node will work as a normal EC node to receive, share, process, store, redirect, or

transmit data packets [33,92]. In addition, this node is able to operate in a passive mode and only analyses the traffic. This attack is considered a passive attack [94,96,97].

- Attackers gain unauthorized access and control of the network, taking advantage of *corrupted or malicious EC nodes*, then inject misleading data packets or can block the delivery of legitimate data packets [92,98–100]. This attack can be launched using three different attack methods: (1) insertion, in which the attacker inserts malicious packets (that seem legitimate) in network communication; (2) manipulation, in which the attacker captures packets, then change them; or (3) replication (or replay), in which the previously exchanged packets between two nodes have been captured and replayed by the attacker.

2. *Side-channel attacks* compromise the security and privacy of users by any accessible information that is not privacy-sensitive in nature, called side-channel information [33,92]. However, this accessible information usually has some correlations with privacy-sensitive data. Attackers explore the hidden correlations and extract the desired sensitive information from side-channel information using specific algorithms or machine learning models. The most popular side channels in EC are: communication signals, electric power consumption, and smartphone/proc filesystem or embedded sensors.

3. *Authentication and authorization attacks* can be categorized into four types [101]: (1) dictionary attacks, in which the attacker utilizes a credential/password dictionary to crack into the authentication of a system [102]; (2) attacks exploiting vulnerabilities in authentication protocols, in which attackers discover the design flaws of the authentication protocols. The most widely adopted authentication protocols in edge computing are Wi-Fi-protected access (WPA/WPA2) and secure sockets layer (SSL)/transport layer security (TLS) protocols; (3) attacks exploiting vulnerabilities in authorization protocols, in which attackers usually exploit the design weaknesses or logic flaws existing in authorization protocols to gain unauthorized access to the sensitive resources or perform privileged operations. The most widely adopted authorization protocol in edge computing is the open authorization (OAuth) protocol [103,104]; (4) Overprivileged attacks, which can happen if an app or a device is granted stronger access rights or more than what is needed.

4. *Jamming attack* is a special type of denial-of-service (DoS) attack. The network will be flooded intentionally by attackers using counterfeit messages to exhaust communication, computing, or/and storage resources. This attack will make authorized users unable to use the infrastructure of the EC-based IoT network [33,92,105].

5. The most famous types of *distributed denial-of-service (DDoS) attacks* [101,106] against EC nodes are outage attacks, sleep deprivation, and battery draining. In outage attacks, EC nodes do not perform their normal operations because of unauthorized access by attackers. In sleep deprivation, attackers overwhelm EC nodes with too many legitimate requests. This kind of attack is very hard to detect. In battery draining, the batteries of sensors, devices, or EC nodes are depleted; therefore, node failure or outage occurs. The most common DDoS attack at the communication level is jamming the transmission of signals, including continuous jamming over all transmissions and intermittent jamming by sending/receiving packets periodically by EC nodes [32,92,98–100,105,107–111].

DDoS attacks in edge computing can be classified as flooding-based attacks and zero-day attacks. In flooding-based attacks (e.g., UDP flooding, SYN flooding, HTTP flooding), the attacker tries to saturate the server and shut down the normal service of a server by flooding with malicious/malformed network packets. A zero-day attack is more difficult to implement. In zero-day DDoS attacks, the attacker should find an unknown vulnerability, i.e., a zero-day vulnerability in the code running on the target edge device/server. These vulnerabilities can trigger memory failure/corruption, resulting in a service shutdown. A zero-day attack is also difficult to defend since it exploits a zero-day unknown vulnerability.

6. *Physical attacks/tampering* happen when attackers can access the EC nodes/devices physically. Then, attackers can extract valuable and sensitive cryptographic data, tamper with the circuit, or modify software/operating systems [33,92].

7. *Eavesdropping or sniffing attacks* occur when adversaries covertly listen to private conversations, such as usernames, passwords, etc., over communication links. Attackers will be able to gain crucial information about the network, for instance, when sniffed packets contain control or access information of the EC nodes, such as configurations and identifiers of nodes or passwords of the shared network [92,98,105].

8. EC nodes can reveal critical information even when they are not transmitting any data as nonnetwork side-channel attacks. For example, the detection of known electromagnetic/acoustic signals or protocols from medical devices can lead to serious privacy issues since critical information about the patient and device can be leaked [8,92].

9. By redirecting, misdirecting, spoofing, or dropping data packets at the communication level, attackers can change routing information and affect how messages are routed [8,92,105]. *Routing information attacks* can appear in different types: (1) altering attack, in which the routing information will be modified by the attacker, for instance, through routing loops or false error messages; (2) blackhole attack [112,113], in which a malicious node attracts all the traffic by advertising the shortest path to the destination, then the attacker processes the packets sent to the malicious node or just drops them; (3) gray hole attack [113] is a kind of blackhole attack in which selective packets will be dropped; (4) worm hole attack, in which the attacker records packets at one network location first, the afterward will tunnel them to another location [114]; (5) hello flood attack, in which the attacker sends "HELLO PACKETS" to all the other nodes using a high transmission power malicious node claiming that it is their neighbor [115,116]; and (6) sybil attack, in which the attacker uses/adds nodes with fake identities called sybil nodes that are able to out-vote genuine nodes in the system [117].

10. The attackers in *forgery attacks* inject new fraudulent data packets and interfere with the receiver, which causes system damage or failure. These data packets can be inserted into communication links using methods such as inserting malicious data packets that seem legitimate, capturing then modifying data packets, and replicating previously exchanged packets between two EC nodes/devices [8,32,92,109].

11. The neighboring EC nodes communicate with each other to access or share data, but every node should only communicate with those nodes that need its data. In *unauthorized control access*, attackers can control the whole neighboring nodes if they gain access to one of the unsecured EC nodes [8,92,118].

12. Two types of *integrity attacks against machine learning* can happen in machine learning methods used in EC-based IoT [92]: (1) causative attack, in which attackers change the training process of machine learning models by manipulating or injecting misleading training data set; and (2) exploratory attack, in which attackers misuse vulnerabilities without changing the training process.

13. In *replay attacks or freshness attacks*, attackers capture and record data traffic for a particular period of time and then use these historical data to replace the current real-time data. This can cause energy and bandwidth consumption of EC nodes and other adverse effects [8,33,92,109].

14. *Insufficient/inessential logging attacks* can damage EC-based IoT systems when log files are not encrypted. System and infrastructure developers must log events such as application errors and attempts of unsuccessful/successful authorization/authentication [32,33,92].

15. *Security threats from/on IoT devices* include mobile botnets, ransomware, and IoT malware [33,92]. In 2017, over 1.5 million attacks were reported that originated from mobile malware [99]. Such threats can lead to data leakage/corruption or even application death [99,105].

16. *Nonstandard frameworks and inadequate testing* and coding flaws are able to cause serious security and privacy attacks [33]. Nodes usually need to be connected to intermediate servers; therefore, compromise could be increased. EC-based system development is a complicated procedure that needs to combine heterogeneous devices/resources created by diverse manufacturers [119]. Moreover, there is no standard framework or policy for the

implementation of EC-based systems. Therefore, some security and privacy flaws of these systems may stay undetected.

17. Functionalities of EC nodes may need to extract personal data from the information generated by user devices. Some of the sensitive information (such as personal activities, preferences, or health status) that must belong to data owners could be shared with other users or network entities without any permission from the data owners, which makes them vulnerable to intruders during data transmission/sharing, causing *privacy leakage*. Location awareness of EC nodes, e.g., Wi-Fi hotspots and base stations (BSs), can be exploited by attackers, and then they can detect and track the device's physical location or other sensitive information from the physical location of EC nodes. In addition, if users connect to multiple EC nodes simultaneously to access a particular service, the physical location of a user's device can be precisely detected using positioning techniques [32,33,99,105].

The EC paradigm in IoT is a combination of heterogeneous resources and devices manufactured by various vendors. Generally, there is no agreed framework or standard policies for the implementation of this paradigm; still, there are many undetected security and privacy threats [33].

Table 1 shows a summary of security and privacy attacks and related solutions and countermeasures according to the solutions explained in Section 4 and their related IDs in Table 2. Table 3 shows the definitions of the security principles outlined in Table 1.

Table 1. Summary of security and privacy attacks and related solutions and countermeasures.

	Against							
	Confidentiality	**Integrity**	**Availability**	**Accountability**	**Nonrepudiation**	**Trust**	**Privacy**	
Attacks and threats								Solutions
Malicious injections	✓	✓	✓	✓	✓	✓	✓	18
Node replication	✓	✓	✓	✓	✓	✓	✓	
Hardware Trojans	✓	✓	✓	✓	✓	✓	✓	4, 5, 6
Camouflage	✓	✓	✓	✓	✓	✓	✓	4, 5, 6, 11
Corrupted or malicious EC nodes	✓	✓	✓	✓	✓	✓	✓	4, 5, 6, 11
Injecting fraudulent packets		✓			✓	✓	✓	10, 11
Side-channel attacks	✓				✓		✓	6
Jamming attacks			✓	✓	✓		✓	2, 8, 10, 11
Denial-of-service (DoS) attacks			✓	✓	✓		✓	1, 2, 8, 10, 11
Physical attacks/tampering	✓	✓	✓	✓	✓	✓	✓	6
Eavesdropping or sniffing	✓				✓		✓	11
Routing information attacks	✓	✓		✓	✓		✓	9
Forgery attacks		✓			✓	✓	✓	10, 11
Unauthorized control access	✓	✓	✓	✓	✓	✓	✓	10, 14
Integrity attacks against machine learning	✓	✓						19
Replay attack or freshness attacks		✓			✓	✓	✓	10, 11
Insufficient/inessential logging attacks	✓			✓	✓		✓	18
Nonstandard frameworks and inadequate testing	✓	✓	✓	✓	✓	✓	✓	18

Table 2. Summary of security and privacy solutions and countermeasures.

Solutions and Countermeasures		**ID**	**Explanation**
Packet filters		1	Accept or deny packets from particular addresses or services by setting up routers, firewalls, or servers [120].
Firewalls		2	Apply a set of rules at the boundary between two or more networks and specify which traffic is allowed and which is denied.
Physical security		3	Limits access to key resources by keeping the resources behind a locked door and/or protected from natural and human-made disasters, intentional and unintentional misuses of equipment, hackers, competitors, and terrorist and biohazard events by keeping resources behind a locked and protected place [120].
Countermeasures for malicious hardware/software injection	Side-channel signal analysis	4	By implementing timing, power, and spatial temperature testing analysis and by detecting unusual behaviors of nodes/devices, detecting hardware Trojans and malicious firmware/software installed on IoT EC nodes/devices.
	Trojan activation methods	5	Compares the outputs, behavior, and side-channel leakages of Trojan-inserted versus Trojan-free circuits in order to detect and model malicious attacks
	Circuit modification or replacing	6	This countermeasure includes: (a) tamper-preventing and/or self-destruction; (b) minimizing information leakage; and (c) PUF into the circuit hardware.
Policy-based mechanisms		7	Ensure that standard rules are not breached; this way, they can detect any violation of policies, and they can detect any abnormal requests to the EC nodes [92].
Securing firmware update		8	The network's firmware can be updated reliably, either remotely or directly. Both methods should have authentication and integrity to ensure secure updates [92].
Reliable routing protocols		9	A table of trusted nodes for sharing sensitive and private information will be created by EC nodes [8,92,98].
Intrusion detection system (IDS)		10	Mitigates security threats using: (1) monitoring network operations and communication links; (2) reporting suspicious activities; and (3) detecting routing attacks and blackhole attacks.
Cryptographic schemes		11	Strong and efficient encryption countermeasure strategies that secure communication protocols against different attacks.
Depatterning data transmissions		12	Prevent side-channel attacks by intentionally inserting fake packets that change the traffic pattern [92,105,108].
Decentralization		13	To ensure anonymity, this mechanism distributes sensitive information among EC nodes in a way that no node has complete knowledge of the information [105].
Authorization		14	Prevents responses to requests originated by attackers or malicious EC nodes. It inspects if an entity can access, control, modify, or share the data [8,92,108,111].
Authentication		15	An action of verifying user identities who request certain services.
Accounting (auditing)		16	Collects network activity data to effectively analyze the security of a network and to respond to security incidents.
Information flooding		17	Prevents intruders from detecting and tracking the location of the information source [98].
Prior testing		18	A behavioral test of the components of the EC network. Conducted prior to the actual operation; performed by applying special inputs, pilot, and/or token signals to the network and monitoring their outputs.
Outlier detection		19	Attacks against machine learning methods inject data outliers into the training data set. These kinds of attacks are drastically mitigated by statistical data analytics methods [98,105].
Secure data aggregation		20	In this scheme, individual devices encrypt their data independently using homomorphic encryption schemes, then send the encrypted data to the EC nodes. EC nodes will aggregate all data, compute the multiplication of individual data, and send the aggregated results to the central cloud servers.
Secure data deduplication		21	Allows the intermediaries to detect the replicate data without learning any knowledge about the data.
Secure data analysis		22	Partitioning functionality execution across edge nodes/devices and the cloud enables individuals that locally and independently train their models and only share their trained models to keep their original data and respective private training set.
Combining EC and blockchain technologies		23	A blockchain provides a trusted, reliable, and secure foundation for information transactions and data regulation between various operating network edge entities based on a consensus mechanism.

Table 3. Security principles.

Fundamental Principles	Definition
Confidentiality	To ensure that information is only available or disclosed to unauthorized individuals, entities, or processes
Integrity	To ensure that information is accurate and complete without any manipulation by unauthorized people
Availability	To ensure that information and services are accessible and usable when requested by an authorized entity
Accountability	An individual is responsible for proper authority for their actions
Nonrepudiation	To be able to prove the occurrence of a claimed event or action
Trust	To be able to provide confidence to others of the qualifications, capabilities, and reliability of that entity to perform specific tasks and fulfill assigned responsibilities
Privacy	To ensure that the confidentiality of, and access to, certain information is protected

4. Security and Privacy Countermeasures and Solutions in EC-Based IoT

This section explains the main strategies and solutions developed to countermeasure the security and privacy attacks and threats explained in the previous section.

1. *Packet filters* protect network resources from unauthorized use, theft, destruction, or DoS attacks. There are two kinds of packet filter policies: one policy denies specific types of packets and accepts all others, and the second one accepts specific types of packets and denies all others.

2. *Firewalls* can be applied as software, hardware appliance, or a router with access control lists (ACLs). Some types of firewalls are: static stateless packet filter firewalls that check packets individually and are optimized for speed and configuration simplicity, stateful firewalls that allow or deny traffic by tracking communication sessions, and proxy firewalls that inspect packets, have support for stateful tracking of sessions, and are able to block malicious traffic or unacceptable content [120].

3. Effective techniques developed to tackle *countermeasures for malicious hardware/software injection* are as follows: (1) side-channel signal analysis, which detects both hardware Trojans by implementing timing, power, and spatial temperature testing analysis, and malicious firmware/software installed on EC nodes/devices by detecting unusual behaviors of nodes/devices, for instance, a significant increase in their heat, execution time, or power consumption [92]; (2) trojan activation methods, which compare the outputs, behavior, and side-channel leakages of Trojan-inserted versus Trojan-free circuits to detect and model malicious attacks [92,105]; and (3) circuit modification or replacing, which is also an effective solution against physical/hardware, Trojan, and side-channel attacks. This countermeasure includes [92]: (a) tamper-preventing and/or self-destruction—to prevent malicious attacks, EC nodes are physically embedded with hardware, or in the worst case, the EC nodes destruct themselves and/or erase their data; (b) minimizing information leakage—in this mechanism, random noise or delay is added to the data intentionally to implement a constant execution path code and to balance Hamming weights; and (c) embedding the physically unclonable function (PUF) into the circuit hardware—this enables device identification and authentication to detect Trojan attacks.

4. *Intrusion detection system (IDS)* is the second line of defense to mitigate security threats by [8,32,92,121]: (1) monitoring network operations and communication links; (2) reporting suspicious activities, for example, when predefined policies are breached or when invalid information is injected into the system; and (3) detecting routing attacks (e.g., spoofing or modification of information) and blackhole attacks.

Wang et al. [122] proposed an IDS architecture for EC-based IoT that integrates a trust evaluation mechanism using the EC platform and service template with balanced dynamics using the EC network. A more general EC IDS architecture has been proposed by Lin et al. [123], showing an efficient, fair resource allocation in EC-based IoT systems.

The hierarchical distributed intrusion detection system (HD-IDS) [124] based on the fog architecture is a hierarchical protection detection that deploys multiple IDSs in different network layers and performs detection by multiple layers collaboratively with traffic analysis; therefore, it provides real time and precise protections. HD-IDS is mainly for detecting traffic injection attacks.

Another approach is called the real-time traffic monitoring system (RTMS) [125], which thoroughly inspects data packets and matches the SQLI pattern in the IDS database to form signature rules, avoiding the workload of manually writing signature rules. The RTMS detects traffic injection attacks more efficiently. This approach mainly updates the traffic injection signature rules through historical attack data analysis, which requires relatively low real-time requirements for rule updates.

5. *Cryptographic schemes* are strong and efficient encryption countermeasure strategies that are used to secure communication protocols against different attacks, such as eavesdropping or routing attacks. There is a wide variety of encryption/decryption strategies that can enhance network security and privacy [126–128], but these solutions are applicable for wired networks. Since EC nodes are usually small sensors with limited resources, e.g., battery power, computing/processing capabilities, and storage memory, employing standard encryption/decryption techniques will cause high memory usage, delay, and power consumption [92,98,105]. The architectures and ideas of several key cryptosystems, such as proxy encryption, attribute-based encryption, searchable encryption, identity-based encryption, and homomorphic encryption, have been explored by Zhang et al. [108]. The work in [129] proposes a secure data-sharing scheme for IoT based on EC smart devices. The proposed scheme uses both public key and secret key encryptions. Moreover, a searching strategy is presented that enables authorized users to perform secure data searches within shared, encrypted, and stored data in IoT based on EC networks without leaking data, secret keys, or keywords. An architecture based on the data proxy concept has been presented in [130] that applies process knowledge to enable security via abstraction and also privacy via remote data fusion.

6. In the EC-based IoT environment, entities are required to be authenticated mutually with one another across different trust domains. There are *authentication* mechanisms such as single/cross-domain and handover authentication. These mechanisms are discussed in detail in [8,32,98,105,108,111,131].

7. *Accounting (auditing)* is collecting network activity data to be able to effectively analyze the security of a network and respond to security incidents. For networks with strict security policies, all attempts to achieve authentication and authorization by any person should be included in audit data. Logging "anonymous" or "guest" access to public servers is especially important. All attempts by users to modify their access rights should also be logged in the data. More information about the collected data in accounting can be found in [120].

A security assessment is a further extension of auditing, in which the network is examined from within by professionals trained in the vulnerabilities exploited by network invaders. Periodic assessments of network vulnerabilities should be part of any security policy and audit procedure. In addition, as a result, a specific plan should be made for correcting deficiencies, which might be as simple as retraining staff [120].

8. *Prior testing* is a behavioral test of the whole and the components of the EC network (e.g., EC routers/nodes, servers, etc.) that is conducted prior to the actual operation. This is performed by applying special inputs, pilot, and/or token signals to the network and monitoring their outputs. The aim of this solution is mainly to identify the possible attacks, simulate them, and evaluate their impacts on the EC-based IoT. It also defines which information must be logged and which is sensitive to be shared or stored [33,92,105,132]. Furthermore, the input files should be inspected closely to prevent any malicious injection.

9. *Secure data aggregation* is a highly secure, privacy-preserving [133,134], and efficient data compression strategy. In this scheme, individual devices encrypt their data independently using homomorphic encryption schemes (such as the Brakerski, Gentry, and Vaikuntanathan (BGV) cryptosystem), then send the encrypted data to the EC nodes. The EC nodes will aggregate all data, compute the multiplication of individual data, and send the aggregated results to the central cloud servers [98,99,105,135].

Discarding the replicate copies of data on intermediaries is required to be able to save the bandwidth in IoT networks. However, this method can disclose sensitive information to intermediaries or intruders. Data encryption is common to protect information and prevent data leakage, but it is not possible to detect replicate copies of data on intermediaries after encrypting the data since all the data will be transformed to random values. To overcome this threat, *secure data deduplication* has been proposed that allows the intermediaries to detect the replicate data without learning any knowledge about the data [99,105].

10. *Secure data analysis*: with the advances in EC technology, some artificial intelligence (AI) functionalities can be shifted from the centralized cloud to EC devices/nodes. This can improve security, privacy, and latency. For instance, partitioning functionality execution across edge nodes/devices and the cloud enables individuals that locally and independently train their models and only share their trained models; therefore, they can keep their original data and respective private training set [99,105]. This reduces the risk of privacy leakage.

11. *Combining EC and blockchain technologies*: a blockchain provides a trusted, reliable, and secure foundation for information transactions and data regulation between various operating network edge entities. Decisions about the correct execution of particular transactions are based on a consensus mechanism without depending on a trusted central authority between the communicating IoT edge nodes [110,136]. In [137], the authors integrated smart contract technologies with a consortium blockchain and developed a secure distributed data storage and sharing method for vehicular EC networks. Gai et al. [138] combined EC and blockchain technologies and proposed a permissioned blockchain EC model to address the privacy-preserving and energy security of smart grid IoT based on EC networks by combing EC and blockchain technologies. A security-aware strategy based on smart contracts running on the blockchain has also been presented in this work.

Table 2 shows a summary of security and privacy solutions and countermeasures. It is worth noting that some of the security and privacy-related concepts, attacks, and solutions explained in the reference papers were in the context of centralized cloud-based IoT, but they are also applicable or can be extended to EC-based IoT.

5. Architecture of Edge Computing-Based IoT

Edge computing in IoT focuses on deploying edge computing into different IoT scenarios to reduce network traffic and decision-making delay.

With the goal of proposing a novel application scenario for edge computing, we present the architecture of edge computing in an IoT-based scenario. As shown in Figure 1, the architecture is divided into three layers: the IoT layer, the edge layer, and the cloud layer, which are derived from the existing edge computing reference architectures [34,35]. The proposed architecture in this paper comprehensively considers the characteristics of IoT and edge computing. We focus on the functions that each layer should have and how layers communicate with each other in detail.

1. *IoT layer:* the IoT layer is formed by all kinds of devices or equipment, handheld terminals, instruments and meters, smart machines, smart vehicles, robots, and other physical objects that monitor services, activities, or equipment in operation. This layer also includes sensors, actuators, controllers, and gateways for IoT environments, allowing for the management and storage of computational resources in IoT devices. These devices or equipment collect a large amount of data by all kinds of sensors using various types of wireless networks, transmit the data to the edge layer, and wait for the control instructions from the edge layer. In IoT environments, a wireless standard, such as Wi-Fi [139], Bluetooth [140], RFID [141], NB-IoT [142–145], 5G [146,147], BLE [18], ZigBee [19,21], LoRa [20], SigFox [139], etc., is normally applied [148].

2. *Edge layer:* this layer is the core layer of the EC-based IoT architecture and is basically responsible for receiving, processing, and forwarding data flow from the IoT layer. The edge layer also provides time-sensitive services such as edge security and privacy protection, intelligent computing, edge data analysis, process optimization, and real-time control.

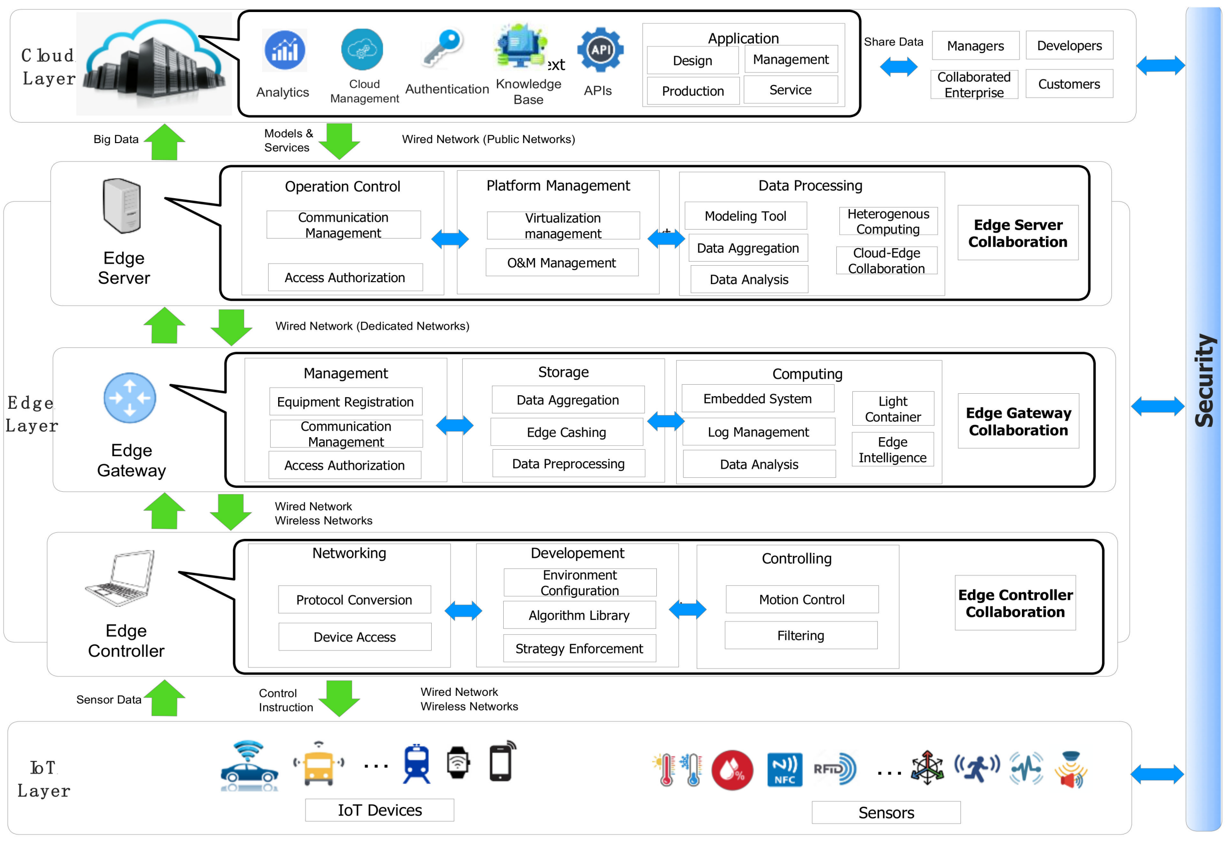

Figure 1. A secure architecture for edge computing-based IoT.

This layer is located close to the edge endpoint nodes; it corrects the orchestration of the different technological assets in the organization and significantly improves the supply, monitoring, and updating of existing resources.

Some of the technical support activities of this layer include: the management of critical business activities, including the management of physical resources, such as sensors, actuators, and controllers, and the study and analysis of large volumes of data in real time.

In this layer, the data that are coming from the IoT layer will be filtered and preprocessed, then will be sent to the cloud layer. The sensors in IoT nodes collect the data using microcontrollers with reduced computational and storage resources. These data will be filtered and preprocessed in computing elements in the edge layer. Low-cost solutions such as microcomputers based on architectures such as Raspberry Pi [22] or Orange Pi (which are based on Raspberry Pi) can be used for these computing elements. These devices are able to process a much greater amount of information than the microcontrollers that are usually used in IoT devices, with the purpose of minimizing energy consumption and executing simple reading logic and control programs on general-purpose input and output ports (GPIO), serial ports such as universal asynchronous receiver-transmitter (UART), serial peripheral interface (SPI), or communication buses between integrated circuits (I^2C).

This kind of device can come with I/O ports allowing to connect them with sensors and actuators; implementation of the edge and IoT layers within a single device is also possible. These devices are used to coordinate a network of cheaper IoT devices that work as hub elements or data collectors throughout the IoT and up to the edge. These devices sometimes must be powered by batteries and solar panels, which are placed beyond the reach of power supply networks.

With the help of these layers, it is possible to apply machine learning techniques to the edge, for example, through TensorFlowLite libraries [22] in Raspberry Pi or similar devices; therefore, it facilitates the performance of data analytics in the edge, saves the amount of data sent to the cloud, reduces associated costs, and offers valuable data to users even if the connection with the Internet and the cloud are temporarily lost.

Based on the data processing capacity of different equipment in the edge layer, the edge layer can be distributed in three sublayers as follows: edge controller layer, edge gateway layer, and edge server layer.

a. *Edge controller layer:* this layer collects data from the IoT layer using some edge controllers, performs preliminary threshold judging or data filtering, and transfers control flow from the edge layer or cloud layer to the IoT layer.

Sensors and devices in the IoT layer are heterogeneous; therefore, the edge controllers in the edge controller layer must be compatible with various protocols and be able to access various sensors or devices, so they will be able to collect data from time delay-sensitive networks of the IoT in real time. After collecting the data from IoT devices, it should be preprocessed for threshold judging or data filtering. Hence, edge controllers of the edge controller layer should integrate the algorithm library based on the environment configuration in order to enhance the effectiveness of the strategy continuously. In addition, after receiving the decision from the edge controller layer or the upper layers, the edge controllers of the edge controller layer should transfer the control flow to the IoT layer via the programmable logic controller (PLC) control or action control module. Moreover, for certain tasks, different edge controllers may need to cooperate.

The latency of the edge controller layer for judgment and feedback is usually very low at the ms-level. This is very important for some time-sensitive applications or emergency situations. In order to reduce the delay or protect life/property, these applications/emergencies must be processed in the edge controller layer.

b. *Edge gateway layer:* this layer mostly contains edge gateways. It collects the data from the edge controller layer using wired networks (such as fieldbus, industrial ethernet, industrial optical fiber, etc.) or wireless networks (such as Wi-Fi, Bluetooth, RFID, NB-IoT, LoRa, 5G, etc.), caches the collected data and provides heterogeneous computing. In addition, edge gateways in this layer transfer control flow from the upper layers (edge server layer or cloud layer) to the edge controller layer and manage the equipment in the edge gateway layer or edge controller layer.

The edge controller layer only performs simple threshold judging or data filtering; however, the edge gateway layer can execute the collected data from IoT devices since it has more storage and computing resources.

After collecting the heterogeneous data by the edge controller layer from IoT devices, these data will first be preprocessed, fused, and cached in the edge gateway layer. After obtaining enough data, the edge gateways of the edge gateway layer will carry out data processing deployed in the embedded system or the lightweight container, data aggregation, and data analysis in either traditional big data analysis or appearing edge intelligence technologies. The data analysis log will be saved for use in the future.

The latency of the edge gateway layer is usually at the s-level or min-level. This layer grants more extensive judgment via merging data from various devices. The management module with many management functions, such as device management, access management, communication management, etc., and the edge gateway collaboration module are also handled in the edge gateway layer, enabling multilayer and multidevice collaboration. These kinds of events have a few seconds or minutes delay and are perfectly handled at the edge gateway layer.

c. *Edge server layer:* the edge server layer has powerful edge servers. This layer performs more complex and critical data processing and based on the data collected from the edge gateway layer by dedicated networks, it creates directional decision instruction. The edge servers in the edge server layer should also have business application management and platform management functions.

The edge servers in the edge server layer are formed into a small computing platform with more powerful storage and computing resources than the equipment of the edge controller and the edge gateway layers. Therefore, the edge server layer is mostly used for a large amount of heterogeneous data processing and operation, reasoning, and training of more precise models, which results in achieving better production scheduling decisions for the edge network. All kinds of resources in the whole edge layer, which require operation and virtualization management functions of the platform, as well as the deployment and scheduling functions of the edge side business application management in order to reasonably allocate resources and reasonably complete and deliver tasks, are also managed by the edge server layer.

More data from different equipment can be analyzed in the edge server layer, achieving process optimization or the best measures taken in a wider area and a longer period of time. This layer usually has an h-level latency.

3. *Cloud layer:* this layer mainly performs massive data mining and seeks optimal resource allocation across an enterprise, a region, or even nationwide. The data in the edge layer will be transferred to the cloud layer through the public network. The edge layer can receive feedback models, services, and business applications offered in the cloud from the cloud layer. At this level, both public (hosting data on commercial servers) and private (corporate data center) cloud services can be used. In this layer, individual application programming interface (API) calls can be activated via executing more complex sets of operations that involve the use of interactive interfaces and are part of the business applications ecosystem. Applications such as product or process design, comprehensive enterprise management, and sale and after-sale services are supported in the cloud layer. Moreover, in the cloud layer, the data can be shared via cloud collaboration between groups of different attributes (e.g., managers, cooperative enterprises, designers, and customers), achieving more pluralistic and deeper data mining. The decision time for such feedback and services in the cloud layer is normally at the day level. The main components of this layer are as follows:

- Analytics: case-based reasoning, machine learning algorithms, and artificial intelligence techniques give greater flexibility in data analysis and visualization capabilities that are required by different business units and operating teams; massive analytics that need more resources and time can be performed at the cloud layer.
- Cloud management: using a storage and administration service, physical or virtual segregation of the stored data according to the tenant, or enabling to track the use of the service by the tenant is possible. Moreover, several tenants can use the service. This is a perfect feature even in private cloud management services. The tenants can be different departments or working groups of a public or private organization.
- Authentication: this can be performed using authorization or a distributed transaction, according to [35].
- Knowledge base: virtual organization of agents and support decision systems based on sensor data can develop a social machine [24]. Cloud-based orchestration, which enables the provisioning, monitoring, and updating of connected technological resources, can complement this component.
- APIs: cloud services can be called through a set of applications (with standard methods, e.g., HTTP, RESTful, XML, or SOAP calls [25]). Using these applications, services can become available via a standard web browser or other HTTP client applications.

Based on every scenario and the main attacks that can occur in that scenario, we should consider relevant solutions and countermeasures discussed in Section 4 for securing the architecture. In the following sections, we describe how security can be achieved based on our scenario.

6. Application of Proposed Architecture

The reference architecture proposed in Section 5 can be suitable for some IoT scenarios. We selected automatic ticketing as a case to discuss how the proposed architecture can be adapted to the IoT scenarios.

Passengers can purchase and validate public transport tickets in various ways, for instance, through an app solution on the passenger's smartphone, contactless payment cards, the transport provider's own card solutions, or buying tickets with cash. In all of these solutions, the passenger needs to do something. Instead, imagine that ticketing automatically happens when passengers are inside the bus or tram without them having to perform any action. Passengers do not need to think about purchasing/validating tickets or the best price option when they want to travel by public transport when ticketing takes place automatically. Faster boarding, no cash handling, and possibly increasing customer satisfaction, as well as discovering customer behavioral patterns on public transport companies, are just a few of the advantages that automatic mobile ticketing solutions can offer [149,150]. The challenge is how, with high accuracy, to determine whether travelers are inside or outside of a public transport and which kind of public transportation travelers are taking.

Inferring the user's state is necessary to detect the mode of transport of a commuter and to enable automated ticketing. Mobile sensor data combined with the reference sensor data of a transport vehicle can be utilized to detect the activity of a person [151–153] or to derive the transportation mode [154–157]. In particular, contextual data, e.g., the location and start time of a journey, can be utilized to reveal significant journeys of passengers [158–163].

This can be achieved by the collection of movement data of a traveler using sensors of the traveler's smartphone in combination with external data sources such as existing bluetooth low energy (BLE)-enabled beacons [164]. BLE technology is the most popular approach to achieving automatic ticketing [165–168]. By employing BLE, it is possible to completely eliminate the traveler's action during check-in and check-out. The traveler's smartphone communicates with either a BLE-enabled on-board device or a device located at a bus stop for seamless check-in/check-out [166–169]. In these solutions, an application residing on the traveler's smartphone triggers a check-in or check-out event when the traveler enters or leaves the proximity of a BLE beacon. Other common solutions mostly rely on GPS or cellular network coordinates to locate the traveler; however, these solutions are not viable for automatic ticketing.

The accurate in-vehicle presence of the traveler is critical for accurate automatic check-in/check-out [165]. It is challenging to establish whether the traveler is indeed inside the public transport vehicle or is located outside somewhere next to the public transport vehicle using GPS technology. Moreover, a GPS is associated with high energy consumption compared with sensor data, such as a gyroscope, magnetometer, or accelerometer [170], and a GPS also depends on an unobstructed view of satellites, which may be difficult to achieve in urban public transportation infrastructure. Therefore, it is less suitable for the real-time in-vehicle presence detection of people. In addition, a GPS provides only static position information and does not capture the constantly changing environment that a passenger is experiencing during travel. Thus, merely considering location data to monitor passengers is inadequate and can be improved by incorporating contextual information from mobile sensors [171].

In our scenario, a group of passengers travels using several modes of public transportation, such as a bus, train, metro, tram, and city boat, within a middle-sized Scandinavian city. Public transportation vehicles are equipped with BLE beacons that detect the smartphone of a passenger in proximity and broadcast their identifiers to the smartphone. All the passengers have a ticketing app installed on their smartphones with a software development kit (SDK) that allows the collection and sending of smartphone sensor data along with contextual information. The raw passenger data are uploaded to a cloud-based database and transformed in an extract, transform, and load (ETL) process, as shown in Figure 2. By analyzing these data and using machine learning solutions, we have planned to define the mode of public transportation and whether the passenger is inside the public transportation vehicle.

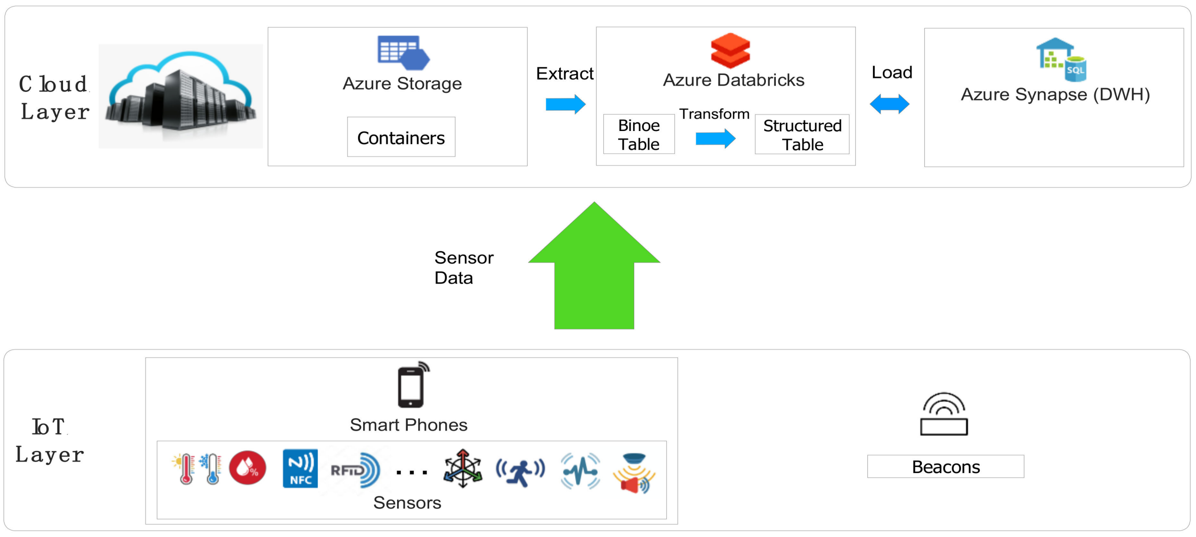

Figure 2. Current cloud-based architecture of the automatic ticketing scenario.

A big challenge here is uploading the data to a cloud-based database that can have high latency because of network traffic or the size of data being uploaded to the cloud, high operational costs, as well as security and privacy risks. To avoid these problems, we can consider three solutions: one solution is performing all the analyses on the mobile phone directly. Another solution is transferring the data to IoT devices, in this case, a Raspberry Pi placed very close to the traveler's smartphone, for instance, inside the public transportation vehicle or at the public transportation stops, then the data will be analyzed on these IoT devices rather than transferring the data to the cloud. Our last proposed solution is implementing edge devices and servers at the cellular base stations.

For our scenario, the IoT layer corresponds to the public transportation, the edge controller layer corresponds to the controller units of smartphones, the edge gateway layer corresponds to the smart gateways, the edge server layer corresponds to the centralized dispatching edge servers in certain cellular base stations, and the cloud layer corresponds to the cloud computing platforms.

As the entity of the IoT layer, the smartphone is equipped with sensors, such as a rotation vector, sounds, a magnetic field, a gyroscope, etc., to collect various sensor data. Meanwhile, the smartphone is equipped with a variety of actuators to execute the instructions from the upper layers.

As the entity of the edge controller layer, the controller units of smartphones are responsible for collecting and recognizing data from different sensors, performing some simple data filtering, data preprocessing and real-time threshold judgment, and transmitting instructions based on the judgment or decision from the upper layers to various action actuators of the IoT layer. For instance, when an error occurs, such as errors in the normal working of a smartphone or in collecting sensor data or full memory errors, the controller units must be able to provide ms-level feedback to the actuators to execute alarms.

The edge gateway layer (in our scenario Raspberry Pis close to the traveler's smartphones) has more storage and computing resources than the edge controller layer and is usually responsible for collecting data from smartphones, preprocessing, fusing, and caching heterogeneous data, and performing corresponding heterogeneous computing. This layer is also responsible for transferring control flow from the upper layers to the edge controller layer. For example, when enough data are obtained from smartphones, Raspberry Pi will perform the steps of data processing, aggregating, and analyzing the data in the way of traditional big data analysis or emerging edge intelligence technologies. The data analysis log will also be saved for future use. The edge gateway layer can monitor and optimize the parameters of different smartphones and identify errors in time. However, if the amount of data is more than the storage capacity of Raspberry Pi or more complex data analysis is required, the data will be transferred to the edge server layer.

As the entity of the edge layer, the area edge server can obtain all the data from Raspberry Pis and perform more comprehensive and complex data analysis, especially when there is a large amount of data, providing some real-time microservices. In addition, the regional edge server must have platform management and application optimization capabilities to manage the Raspberry Pis in the region to make reasonable and effective use of their storage and computing resources and to implement process optimization based on big data.

As the entity of the cloud layer, the cloud service platform obtains the data from edge servers in different base stations and uses it for some top-level decisions and applications, such as process optimization, etc.

In the next section, we will discuss in detail the benefits and disadvantages of these approaches.

6.1. Major Security Threats and Attacks

We summarize the major security threats and attacks faced by edge computing in our scenario. Then, we explore the corresponding defense mechanisms. In the next section, we outline the main causes of the attacks and discuss the practicality of launching/defending them.

1. *DDoS attacks and defense mechanisms*: DDoS attacks are the most common and easiest-to-exploit attacks in the practical world. Therefore, they are significant threats to EC services as well. This attack is very dangerous, especially in scenarios where many IoT devices are involved, such as in our scenario. Lack of suitable security or any vulnerability in IoT devices (mobile phones, Raspberry Pis, and beacons in our scenario) can be misused by attackers to launch a DDoS attack.

DDoS attacks targeting edge computing can be categorized as *flooding-based attacks* and *zero-day attacks*. Flooding-based attacks aim to shut down the normal service of a server using a large amount of flooded malformed/malicious network packets. Zero-day DDoS attacks are more advanced than flooding-based DDoS, but they are more difficult to implement. In such an attack, an unknown vulnerability in a piece of code running on the target edge server/device must be fined by an attacker (i.e., zero-day vulnerability), which can result in memory corruption and a service shutdown.

The detection of flooding-based DDoS attacks can be mainly classified into two categories: per-packet-based detection and statistics-based detection. Per-packet-based detections detect flooding-based attacks at the packet level. A flooding-based DDoS attack is launched mainly by sending an enormous amount of malicious/malformed network packets; therefore, detecting and filtering those packets is an effective defense. Statistics-based approaches mainly detect DDoS attacks based on the advent of clusters of DDoS traffic. The existing statistics-based detection solutions use either packet entropy or machine learning tools.

As per-packet-based detections were used in our scenario, we used packet filters and firewalls against the flooding-based attacks. In addition, for statistics-based detections, we used IDSs. Moreover, it is important to use secure firmware updates. We also considered static analysis techniques to detect any vulnerability in our codes.

2. *Side-channel attacks and defense mechanisms*: defenses against side-channel attacks can be performed by either restricting access to side-channel information or protecting sensitive data from inference attacks.

In our scenario, we considered defense mechanisms such as data perturbation-based techniques (e.g., differential privacy) and source code-level obfuscation and hardware protection to directly manipulate or restrict access to the side channels in order to provide protection.

3. *Malware injection attacks and defense mechanisms*: malware injection attacks in edge computing can be classified into two categories: server-side injections (injection attacks targeting edge servers) and device-side injections (injection attacks targeting edge devices).

The defenses against server-side injections are mainly based on the detection filter philosophy, while code-level analysis for fine-grained access control and malicious behavior detection is mainly used to defend against device-side injections.

To detect these attacks, we considered employing code checking with various schemes such as static analysis, dynamic debugging, blackbox testing, and taint-based analysis; and to prevent any illegal SQL queries from being executed, we considered setting up a proxy filter and employing instruction-set randomization (ISR).

Malware injections attacking mobile devices exploit the design flaws of the mobile OSes and the usage of malicious libraries. We used static analysis methods to identify possible malicious uses of dangerous Android APIs. In addition, we considered using cryptographic techniques, such as signature verification, as well as Trojan activation methods, circuit modification/replacing, and prior testing.

Authentication and authorization attacks and defense mechanisms: other attacks that are very common are authentication and authorization attacks [101]. These attacks can be taxonomized into four types: dictionary attacks, attacks exploiting vulnerabilities in authentication protocols, attacks exploiting vulnerabilities in authorization protocols, and overprivileged attacks. The first two target authentication protocols, and the rest target authorization protocols. In dictionary attacks, an attacker employs a credential/password dictionary to crack the credential-enabled authentication system.

The defenses against dictionary attacks mainly focus on adding a stronger authentication layer (e.g., two-factor authentication) or hardening the password verification processes, and the defenses against the other three types of attacks mainly use the philosophy of patching/strengthening the current protocols or conducting code-level analyses. Defending against attacks exploiting the vulnerabilities in authentication protocols should either enhance the security of the communication protocols or secure the cryptographic implementations.

Hardening authorization protocols: authorization in an edge computing infrastructure is as important as the authentication system. Without a properly designed authorization mechanism, an attacker can exploit the weaknesses in the authorization system as well. OAuth 2.0 is a secure protocol that can be used. A common approach to defend against overprivileged attacks against smartphone devices is strengthening the current permission models adopted by mobile OSes.

7. Discussion

Cloud computing has been considered one of the major computing paradigms in the field of information technology. This technological strategy offers consumers a smooth communication connection to a system of huge, virtualized computing resources that can be easily reconfigured for the scalable demands of users. Users can store their data in the cloud and access/retrieve these data when they require it [172]. Using cloud computing, costs of servicing and resource management involved with utilization have been deducted [173]. The cloud providers are responsible for maintaining and managing information over the cloud storage. Despite all the advantages that cloud computing brings for its users, several concerns have been raised: high latency for some time-sensitive applications, as well as security and privacy of the data being transferred to the cloud, to name a few. These are especially problematic in our scenario of automatic ticketing, in which a substantial amount of sensitive real-time information is being collected from the traveler's smartphone. In addition, these data should be analyzed with the least latency to define the location of travelers as well as the mode of public transportation.

To combat the mentioned problems of cloud computing, EC solutions have been proposed. In EC, computations will be performed on the edge of the network close to the smartphones of passengers in our scenario. Placing computation power close to the edge reduces latency occurring during the data transition to/from the cloud and increases the speed of the transaction. Analyzing the data locally in EC protects the information of users from security and privacy vulnerabilities related to the network or the closed system of a service provider and, as a result, reduces the concerns related to sensitive data of a traveler's smartphone. The flow of data into central systems can be optimized, and the volume of raw data can be retained at the edge; therefore, bandwidth and related costs will be reduced. This is beneficial for automatic ticketing services and can help public transportation companies to offer lower prices for their services. The data have been stored and processed locally by edge devices and centers; this can overcome any intermittent connectivity issues.

The first EC solution (mentioned in the previous section) of processing and analyses of all the sensor data at the extreme edge (the smartphones of passengers themselves) would not be possible practically because smartphones have limited computation power, limited memory size, and limited battery life. Thus, we considered uploading the raw data from the passenger's smartphone to other powerful IoT devices such as Raspberry Pi. The second solution, to a great extent, would reduce latency, operational costs, and security and privacy concerns related to cloud-based solutions, as well as intermittent connectivity issues. However, the limited computational power and storage of Raspberry Pi compared with cloud computing servers can be an issue, especially if we have a huge, increased amount of data to process; in this case, Raspberry Pi can communicate with the cloud-based environment to access more computational and storage resources that would cause the same problems of cloud-based computing solutions. Transferring the data to cellular base stations equipped with edge servers that have more powerful storage and computation resources, as well as a more secure environment, would be the best proposed solution. The edge layer can send the data to the cloud to support applications such as comprehensive enterprise management, product/process design, and further services, and gives feedback models and microservices to the edge layer. Moreover, the data can be shared through cloud collaboration among managers, cooperative enterprises, designers, or passengers to achieve more pluralistic and deeper data value mining.

7.1. Major Security Threats and Attacks

1. *DDoS attacks and defense mechanisms*: protocol-level flaws caused by a lack of sufficient security in the initial design are the main reason for flooding-based DDoS attacks. On the other hand, the main reason for zero-day flooding attacks is code-level vulnerabilities. Flooding-based attacks are easy to launch since the attacker only needs to create a large number of malicious packets, which can be provided through compromised distributed devices. However, zero-day attacks may not be very easy and common to launch since discovering zero-day vulnerabilities in a system demands highly sophisticated analysis. Nevertheless, launching a zero-day attack is still quite practical since it is difficult to avoid code-level vulnerabilities, especially in large systems with millions of lines of code. It is even harder to avoid such vulnerabilities in edge computing since edge devices are not usually built with strong security software to avoid high costs and have a better user experience.

Countermeasures against DDoS attacks are still very limited. Per-packet-based detections (e.g., packet filters, firewalls, or some types of IDSs) against flooding-based attacks can be either bypassed using more advanced DDoS attacks that, for instance, exploit address spoofing or require maintaining a large list of authorized IP addresses that might need frequent changes. On the other hand, statistics-based detections (e.g., IDSs) can identify DDoS attacks only when groups of DDoS attacks have been sent to the target edge servers, which can cause irreparable damages. Zero-day DDoS attacks are even harder to detect and defend. Most of the offline detection countermeasures are not able to recognize the exact type or location of the vulnerabilities, and if the DDoS attack shellcodes are encrypted or intentionally modified, most of the real-time online defense mechanisms are barely able to discover the attacks.

2. *Side-channel attacks and defense mechanisms*: the main reason for side-channel attacks is hidden connections between publicly available side-channel data and sensitive data

that should be protected [101]. Leaking of side-channel information is very common and unavoidable. With the advances in machine learning and deep learning, achieving a successful side-channel attack is becoming easier. Research shows that all kinds of side-channel attacks can be practically launched in edge computing [174].

Side-channel attacks are very difficult to defend since they can be launched silently and passively. Defense mechanisms such as data perturbation-based techniques (e.g., differential privacy) can be very effective and prevent attackers from deriving a user's sensitive data, but they may sacrifice the utility of the data [175], while there is still a tradeoff between privacy protection and data utility for some particular applications.

In addition, defense mechanisms such as source code-level obfuscation and hardware protection directly manipulate or restrict access to side channels in order to provide protection; however, it is not possible to use these solutions for any piece of side-channel information, as most of this information is undetectable. Moreover, many of the existing countermeasures are still vulnerable to side-channel attacks [176,177].

3. *Malware injection attacks and defense mechanisms*: the main reason for server-side injections is protocol design flaws, while device-side injections mostly happen because of code-level design flaws, as well as the adoption of device-level coarse-grained access control [101]. Server-side injections are quite common and have been widely exploited in the industry. Even though device-side injections are not as common as those at the server-side, research shows that they can still be launched in practice.

The defense mechanisms against server-side injections are mainly based on the detection filter philosophy, while code-level analysis for fine-grained access control and malicious behavior detection is mainly used to defend against device-side injections. The defense mechanisms against edge device malware injection/modification are not very successful. There are no mature countermeasures defending against zero-day injections, firmware modification attacks, and remote WebView infections. In addition, current solutions such as code-level static analysis provide actions that cannot be used in real time; therefore, they cannot prevent damage; moreover, they need full access to the firmware or source codes, which may not be possible all the time. Some other defenses that are based on weak signature verification are also ineffective. Therefore, defending against malware injection attacks, especially on the edge device side, is still quite challenging.

Authentication and authorization attacks and defense mechanisms: attacks such as brute-force attacks can be easily launched in edge computing systems with little or a reasonable amount of effort. The main reason for brute-force attacks is credential weaknesses in authentication protocols that can be defended by adding a stronger layer of authentication or hardening the password verification processes. Other attacks, such as overprivileged attacks, occur because of protocol/implementation-level flaws and can be defended by code-level analyses or patching/strengthening the current protocols. Strengthening authentication security in edge computing systems is critical, as it is the security entry point of a system. Most attacks begin with breaking into the authentication of systems in the first place.

Defending against dictionary and brute-force attacks might be challenging. Attackers may have their own resources to build their dictionaries/passwords that can have great data to break the authentication system. It has been proven that mechanisms such as two-factor authentication are not effective and can be broken practically. Attacks exploiting the weaknesses of authentication/authorization protocols are not easy and need more effort since the attacker should identify the vulnerabilities of the protocol and compromise the edge servers, but by cracking an authentication/authorization protocol, attackers can break into the system and then begin other attacks. Overprivileged attacks are quite common in IoT and mobile devices, and there are many real-world overprivileged apps [178,179]. However, the corresponding countermeasures are generally not very effective.

8. Challenges of EC-Based IoT

As we discussed, there are numerous benefits to integrating edge computing to assist IoT. In this section, we discuss the main challenges of EC-based IoT.

1. *System integration*: a significant challenge in the edge computing environment is supporting various kinds of IoT devices and different service demands. Edge computing is a heterogeneous system and combines various platforms, network topologies, and servers. Therefore, programming and managing resources and data for diverse applications running on varying and heterogeneous platforms in different locations would be difficult. Another challenge is deploying and managing the huge number of server-side programs on the edge nodes.

In addition, since various storage servers are running with various operating systems, file naming, resource allocation, file reliability management, etc., will be another challenge. Moreover, the naming of data resources becomes another big challenge because of the massive number of IoT devices generating and uploading data at the same time.

2. *Resource management*: because of decentralized resources in edge computing, the significant heterogeneity of service providers, devices, and applications creates substantial complexity.

3. *Security and privacy*: edge computing consists of a complex interweaving of multiple and varied technologies (peer-to-peer systems, wireless networks, virtualization, etc.) and adopting a comprehensive, integrated system to safeguard and manage each technology platform and the system as a whole is required. The security and privacy of distributed structures are very challenging. With respect to privacy, privacy-sensitive information associated with end users could be exploited during data processing at the edge.

With respect to security, one of the security problems of edge computing is authenticating gateways on different levels. Different edge nodes are managed by different owners, so deploying an equivalent security strategy throughout would be difficult. Security in data sharing and data transmission processes are also other key challenges for EC-based IoT. To be able to perform many tasks with limited resources and a huge number of edge devices, multiple devices should collaborate, and data need to be shared securely between edge devices. In EC-based IoT, massive data are generated by numerous sensors and devices, and different third-party suppliers provide all the storage. User data are outsourced to those storage suppliers, whose storage devices are deployed at the edge of the network and located at many different physical addresses. This increases the risk of attacks for several reasons: first, ensuring data integrity is difficult; the data are separated into many parts and are stored across different storage locations, so it is easy to lose data packets or store incorrect data. Second, unauthorized users or adversaries may modify or abuse the uploaded data in storage, which will lead to data leakage and other privacy issues. Another important security challenge in EC-based IoT is maintaining security and privacy in uploading computational tasks to edge computation nodes.

Blockchain can provide security for edge data sharing to some extent. However, since the computing resources of edge devices are usually limited, the design and optimization of edge IoT architecture based on blockchain are a great challenge, and more attention should be paid to the direction of the edge IoT based on blockchain.

4. *Advanced communication*: with advances in communication technologies of future 5G cellular networks, including ultradense networks (UDNs), massive multiple input and multiple output (MIMO), and millimeter-wave, further progress in edge computing becomes inevitable with the integration of these technologies.

The challenges to applying 5G, as the next generation communication technology, in edge computing in IoT focus on QoS, nodes management, and network slicing. The QoS standards and schemes are different in 5G, and edge computing and common problems need to be considered. A large number of 5G base stations helps to collect and manage edge node information, but a suitable edge node management scheme should be carried out according to local conditions. Multiple services can be deployed in parallel because of the network slicing feature of 5G, but the architectures of different systems still need to be tailored according to the actual situation. Furthermore, the hardware transformation of traditional equipment and the remote maintenance schemes for 5G infrastructure also need attention.

5. *Data offloading and load balancing* are always very challenging in EC-based IoT systems because of the huge number of devices, and the computing resources that need to

be scheduled are too distributed. Therefore, data offloading and load balancing schemes should be designed for special requirements.

6. *Edge artificial intelligence* provides new opportunities and challenges for data processing at the edge of the system. The weak computing power of edge devices makes it difficult to complete a large amount of computation quickly on edge devices. Moreover, the model of edge artificial intelligence is usually complex and requires more computing resources to complete the training and inference of the model.

9. Conclusions

In this article, we provided an overview of what edge computing is and what its relationship/difference is with other similar computing paradigms, such as fog computing, cloud computing, and others. We also extracted and summarized the major security and privacy attacks in EC-based IoT, as reported in the literature, and the corresponding countermeasures and solutions that can be applied to these attacks. Next, we proposed a secure EC-based architecture for IoT infrastructure through relevant research achievements concerning edge computing in IoT. In addition, we defined an IoT scenario related to transport mode detection based on cloud computing, then presented two scenarios based on edge computing for the defined IoT scenario and discussed the advantages/disadvantages of scenarios based on edge computing and the scenario based on cloud computing. Last, we discussed the most influential attacks that can occur as well as their corresponding defense mechanisms that can be practically used in our scenario.

Author Contributions: Conceptualization, E.F.; methodology, E.F.; investigation, E.F.; data curation, E.F.; writing—original draft preparation, E.F.; writing—review and editing, E.F.; visualization, E.F.; supervision, T.-M.G.; funding acquisition, T.-M.G. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported by the Norwegian Research Council.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: It has been a project together with Fluxloop company.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Mell, P.; Grance, T. *The Nist Definition of Cloud Computing*; National Institute of Standards and Technology: Gaithersburg, MD, USA, 2011.
2. Dillon, T.; Wu, C.; Chang, E. Cloud computing: Issues and challenges. In Proceedings of the 2010 24th IEEE International Conference on Advanced Information Networking and Applications (AINA), Perth, WA, Australia, 20–23 April 2010; pp. 27–33.
3. Peng, M.; Zhang, K. Recent advances in fog radio access networks: Performance analysis and radio resource allocation. *IEEE Access* **2016**, *4*, 5003–5009. [CrossRef]
4. Al-Fuqaha, A.; Guizani, M.; Mohammadi, M.; Aledhari, M.; Ayyash, M. Internet of things: A survey on enabling technologies, protocols, and applications. *IEEE Commun. Surv. Tutor.* **2015**, *17*, 2347–2376. [CrossRef]
5. Chiang, M.; Zhang, T. Fog and IoT: An overview of research opportunities. *IEEE Internet Things J.* **2016**, *3*, 854–864. [CrossRef]
6. Ganz, F.; Puschmann, D.; Barnaghi, P.; Carrez, F. A practical evaluation of information processing and abstraction techniques for the internet of things. *IEEE Internet Things J.* **2015**, *2*, 340–354. [CrossRef]
7. Linthicum, D. Responsive data architecture for the Internet of Things. *Computer* **2016**, *49*, 72–75. [CrossRef]
8. Lin, J.; Yu, W.; Zhang, N.; Yang, X.; Zhang, H.; Zhao, W. A survey on Internet of Things: Architecture, enabling technologies, security and privacy, and applications. *IEEE Internet Things J.* **2017**, *4*, 1125–1142. [CrossRef]
9. Stankovic, J.A. Research directions for the Internet of Things. *IEEE Internet Things J.* **2014**, *1*, 3–9. [CrossRef]
10. Wu, J.; Zhao, W. Design and realization of winternet: From net of things to Internet of Things. *ACM Trans. Cyber-Phys. Syst.* **2016**, *1*, 1–12. [CrossRef]
11. Lin, J.; Yu, W.; Yang, X.; Yang, Q.; Fu, X.; Zhao, W. A real-time en-route route guidance decision scheme for transportation-based cyberphysical systems. *IEEE Trans. Veh. Technol.* **2017**, *66*, 2551–2566. [CrossRef]
12. Yan, Y.; Qian, Y.; Sharif, H.; Tipper, D. A survey on cyber security for smart grid communications. *IEEE Commun. Surv. Tutor.* **2012**, *14*, 998–1010. [CrossRef]

13. Lin, J.; Yu, W.; Yang, X. Towards multistep electricity prices in smart grid electricity markets. *IEEE Trans. Parallel Distrib. Syst.* **2016**, *27*, 286–302. [CrossRef]
14. Mohamed, N.; Al-Jaroodi, J.; Jawhar, I.; Lazarova-Molnar, S.; Mahmoud, S. SmartCityWare: A service-oriented middleware for cloud and fog enabled smart city services. *IEEE Access* **2017**, *5*, 17576–17588. [CrossRef]
15. Mallapuram, S.; Ngwum, N.; Yuan, F.; Lu, C.; Yu, W. Smartcity: The state of the art, datasets, and evaluation platforms. In Proceedings of the 2017 IEEE/ACIS 16th International Conference on Computer and Information Science (ICIS), Wuhan, China, 24–26 May 2017; pp. 447–452.
16. Ciaetal, M.D. Usingsmartcitydatain5Gself-organizingnetworks. *IEEE Internet Things J.* **2017**, *5*, 645–654.
17. Mao, Y.; You, C.; Zhang, J.; Huang, K.; Letaief, K.B. A survey on mobile edge computing: The communication perspective. *IEEE Commun. Surv. Tutor.* **2017**, *19*, 2322–2358. [CrossRef]
18. Cho, K.; Park, G.; Cho, W.; Seo, J.; Han, K. Performance analysis of device discovery of bluetooth low energy (BLE) networks. *Comput. Commun.* **2016**, *81*, 72–85. [CrossRef]
19. Montori, F.; Bedogni, L.; Di Felice, M.; Bononi, L. Machine-to-machine wireless communication technologies for the internet of things: Taxonomy, comparison and open issues. *Pervasive Mob. Comput.* **2018**, *50*, 56–81. [CrossRef]
20. LoRa-Alliance, A Technical Overview of LoRa and Lo-RaWAN What Is It? Technical Report. Available online: https://lora-developers.semtech.com/uploads/documents/files/LoRa_and_LoRaWAN-A_Tech_Overview-Downloadable.pdf (accessed on 20 November 2018).
21. García, O.; Alonso, R.S.; Prieto, J.; Corchado, J.M. Energy efficiency in public buildings through context-aware social computing. *Sensors* **2017**, *17*, 826. [CrossRef]
22. Tang, J. *Intelligent Mobile Projects with TensorFlow: Build 10+ Artificial Intelligence Apps Using TensorFlow Mobile and Lite for iOS, Android, and Raspberry Pi*; Packt Publishing: Birmingham, UK, 2018.
23. Das, A.; Patterson, S.; Wittie, M. Edgebench: Benchmarking edge computing platforms. In Proceedings of the 2018 IEEE/ACM International Conference on Utility and Cloud Computing Companion (UCC Companion), Zurich, Switzerland, 17–20 December 2018; pp. 175–180.
24. González-Briones, A.; Chamoso, P.; Yoe, H.; Corchado, J.M. GreenVMAS: Virtual organization based platform for heating greenhouses using waste energy from power plants. *Sensors* **2018**, *18*, 861. [CrossRef]
25. Alonso, R.S.; Tapia, D.I.; Bajo, J.; García, O.; de Paz, J.F.; Corchado, J.M. Implementing a hardware-embedded reactive agents platform based on a service-oriented architecture over heterogeneous wireless sensor networks. *Ad Hoc Netw.* **2013**, *11*, 151–166. [CrossRef]
26. Khan, W.Z.; Ahmed, E.; Hakak, S.; Yaqoob, I.; Ahmed, A. Edge computing: A survey. *Future Gener. Comput. Syst.* **2019**, *97*, 219–235. [CrossRef]
27. Hsu, R.; Lee, J.; Quek, T.Q.S.; Chen, J. Reconfigurable security: Edge-computing-based framework for IoT. *IEEE Netw.* **2018**, *32*, 92–99. [CrossRef]
28. Sha, K.; Errabelly, R.; Wei, W.; Yang, T.A.; Wang, Z. EdgeSec: Design of an edge layer security service to enhance IoT security. In Proceedings of the 2017 IEEE 1st International Conference on Fog and Edge Computing (ICFEC), Madrid, Spain, 14–17 May 2017; pp. 81–88.
29. Condry, M.W.; Nelson, C.B. Using smart edge IoT devices for safer, rapid response with industry IoT control operations. *Proc. IEEE* **2016**, *104*, 938–946. [CrossRef]
30. King, J.; Awad, A.I. A distributed security mechanism for resource-constrained IoT devices. *Informatica* **2016**, *40*, 663–667.
31. Yousefpour, A.; Fung, C.; Nguyen, T.; Kadiyala, K.; Jalali, F.; Niakanlahiji, A.; Kong, J.; Jue, J.P. All one needs to know about fog computing and related edge computing paradigms: A complete survey. *J. Syst. Archit.* **2019**, *98*, 289–330. [CrossRef]
32. Roman, R.; Lopez, J.; Mambo, M. Mobile edge computing, Fog et al.: A survey and analysis of security threats and challenges. *Future Gener. Comput. Syst.* **2018**, *78*, 680–698. [CrossRef]
33. Alwarafy, A.; Al-Thelaya, K.A.; Abdallah, M.; Schneider, J.; Hamdi, M. A survey on security and privacy issues in edge-computing-assisted Internet of Things. *IEEE Internet Things J.* **2020**, *8*, 4004–4022. [CrossRef]
34. Qiu, T.; Chi, J.; Zhou, X.; Ning, Z.; Atiquzzaman, M.; Wu, D.O. Edge computing in industrial internet of things: Architecture, advances and challenges. *IEEE Commun. Surv. Tutor.* **2020**, *22*, 2462–2488. [CrossRef]
35. Sittón-Candanedo, I.; Alonso, R.S.; Corchado, J.M.; Rodríguez-González, S.; Casado-Vara, R. A review of edge computing reference architectures and a new global edge proposal. *Future Gener. Comput. Syst.* **2019**, *99*, 278–294. [CrossRef]
36. Vaquero, L.M.; Rodero-Merino, L.; Caceres, J.; Lindner, M. A break in the clouds: Towards a cloud definition. *ACM Sigcomm Comput. Commun. Rev.* **2008**, *39*, 50–55. [CrossRef]
37. OpenFogConsortium, Openfog Reference Architecture for Fog Computing. Available online: https://www.openfogconsortium.org/ra/ (accessed on 15 February 2017).
38. Bonomi, F.; Milito, R.; Zhu, J.; Addepalli, S. Fog computing and its role in the internet of things. In Proceedings of the First Edition of the MCC Workshop on Mobile Cloud Computing; ACM: New York, NY, USA, 2012; pp. 13–16.
39. Satyanarayanan, M. Fundamental challenges in mobile computing. In Proceedings of the Fifteenth Annual ACM Symposium on Principles of Distributed Computing; ACM: New York, NY, USA, 1996; pp. 1–7.
40. Dinh, H.T.; Lee, C.; Niyato, D.; Wang, P. A survey of mobile cloud computing: Architecture, applications, and approaches. *Wirel. Commun. Mob. Comput.* **2013**, *13*, 1587–1611. [CrossRef]

41. OpenEdgeConsortium, About—The Who, What, and How. Technical Report, OpenEdge Computing. Available online: http://openedgecomputing.org/about.html (accessed on 15 February 2022).
42. Reale, A. A Guide to Edge IoT Analytics. Blog, International Business Machines. 2017. Available online: https://www.ibm.com/blogs/internet-of-things/edge-iot-analytics (accessed on 23 February 2017).
43. Mach, P.; Becvar, Z. Mobile edge computing: A survey on architecture and computation offloading. *IEEE Commun. Surv. Tutor.* **2017**, *19*, 1628–1656. [CrossRef]
44. Garcia Lopez, P.; Montresor, A.; Epema, D.; Datta, A.; Higashino, T.; Iamnitchi, A.; Barcellos, M.; Felber, P.; Riviere, E. Edge-centric computing: Vision and challenges. *ACM SIGCOMM Comput. Commun. Rev.* **2015**, *45*, 37–42. [CrossRef]
45. What Is Edge Computing? Blog, General Electric. Available online: https://www.ge.com/digital/blog/what-edge-computing (accessed on 15 February 2022).
46. Satyanarayanan, M.; Bahl, P.; Caceres, R.; Davies, N. The case for vm-based cloudlets in mobile computing. *IEEE Pervasive Comput.* **2009**, *8*, 14–23. [CrossRef]
47. Bahl, V. The Emergence of Micro Datacenters (Cloudlets) for Mobile Computing. 2015. Available online: https://pdfs.semanticscholar.org/e393/e4f5ffa0fa23a6ea8686c7461463a6bb31b8.pdf (accessed on 13 May 2015).
48. Hao, P.; Bai, Y.; Zhang, X.; Zhang, Y. Edgecourier: An edge-hosted personal service for low-bandwidth document synchronization in mobile cloud storage services. In *Proceedings of the Second ACM/IEEE Symposium on Edge Computing*; ACM: New York, NY, USA, 2017; pp. 1–14.
49. Cui, Y.; Song, J.; Ren, K.; Li, M.; Li, Z.; Ren, Q.; Zhang, Y. Software defined cooperative offloading for mobile cloudlets. *IEEE/ACM Trans. Netw.* **2017**, *25*, 1746–1760. [CrossRef]
50. Drolia, U.; Martins, R.; Tan, J.; Chheda, A.; Sanghavi, M.; Gandhi, R.; Narasimhan, P. The case for mobile edge-clouds. In Proceedings of the 2013 IEEE 10th International Conference on Ubiquitous Intelligence and Computing and 2013 IEEE 10th International Conference on Autonomic and Trusted Computing, Vietri sul Mare, Italy, 18–21 December 2013; pp. 209–215.
51. Shi, C.; Lakafosis, V.; Ammar, M.H.; Zegura, E.W. Serendipity: Enabling remote computing among intermittently connected mobile devices. In Proceedings of the thirteenth ACM international symposium on Mobile Ad Hoc Networking and Computing, Hilton Head Island, SC, USA, 11–14 June 2012; pp. 145–154.
52. Mtibaa, A.; Fahim, A.; Harras, K.A.; Ammar, M.H. Towards resource sharing in mobile device clouds: Power balancing across mobile devices. *ACM SIGCOMM Comput. Commun. Rev.* **2013**, *43*, 51–56. [CrossRef]
53. Mtibaa, A.; Harras, K.A.; Fahim, A. Towards computational offloading in mobile device clouds. In Proceedings of the 2013 IEEE 5th International Conference on Cloud Computing Technology and Science, Bristol, UK, 2–5 December 2013; pp. 331–338.
54. Nishio, T.; Shinkuma, R.; Takahashi, T.; Mandayam, N.B. Service-oriented heterogeneous resource sharing for optimizing service latency in mobile cloud. In Proceedings of the First International Workshop on Mobile Cloud Computing & Networking, Bengaluru, India, 29 July 2013; pp. 19–26.
55. Habak, K.; Ammar, M.; Harras, K.A.; Zegura, E. Femto clouds: Leveraging mobile devices to provide cloud service at the edge. In Proceedings of the 2015 IEEE 8th international conference on cloud computing, New York, NY, USA, 27 June–2 July 2015; pp. 9–16.
56. Liu, F.; Shu, P.; Jin, H.; Ding, L.; Yu, J.; Niu, D.; Li, B. Gearing resource-poor mobile devices with powerful clouds: Architectures, challenges, and applications. *IEEE Wirel. Commun.* **2013**, *20*, 14–22.
57. Yaqoob, I.; Ahmed, E.; Gani, A.; Mokhtar, S.; Imran, M.; Guizani, S. Mobile ad hoc cloud: A survey. *Wirel. Commun. Mob. Comput.* **2016**, *16*, 2572–2589. [CrossRef]
58. Ragona, C.; Granelli, F.; Fiandrino, C.; Kliazovich, D.; Bouvry, P. Energy-efficient computation offloading for wearable devices and smartphones in mobile cloud computing. In Proceedings of the 2015 IEEE Global Communications Conference (GLOBECOM), San Diego, CA, USA, 6–10 December 2015; pp. 1–6.
59. Zhang, W.; Wen, Y.; Wu, J.; Li, H. Toward a unified elastic computing platform for smartphones with cloud support. *IEEE Netw.* **2013**, *27*, 34–40. [CrossRef]
60. Hubaux, J.-P.; Gross, T.; Le Boudec, J.-Y.; Vetterli, M. Toward self-organized mobile ad hoc networks: The terminodes project. *IEEE Commun. Mag.* **2001**, *39*, 118–124. [CrossRef]
61. Li, Y.; Wang, W. Can mobile cloudlets support mobile applications? In Proceedings of the IEEE INFOCOM 2014-IEEE Conference on Computer Communications, Toronto, ON, Canada, 27 April–2 May 2014; pp. 1060–1068.
62. Tseng, Y.C.; Ni, S.Y.; Chen, Y.S.; Sheu, J.P. The broadcast storm problem in a mobile ad hoc network. *Wirel. Netw.* **2002**, *8*, 153–167. [CrossRef]
63. Chiang, M.; Ha, S.; Chih-Lin, I.; Risso, F.; Zhang, T. Clarifying fog computing and networking: 10 questions and answers. *IEEE Commun. Mag.* **2017**, *55*, 18–20. [CrossRef]
64. Available online: https://www.cisco.com/c/en/us/solutions/computing/what-is-edge-computing.html#~{}types-of-edge-computing (accessed on 15 February 2022).
65. Acharya, J.; Gaur, S. Edge compression of gps data for mobile iot. In Proceedings of the 2017 IEEE Fog World Congress (FWC), Santa Clara, CA, USA, 30 October–1 November 2017; pp. 1–6.
66. Zhang, T. Fog Computing Brings New Business Opportunities and Disruptions. IoT-Agenda/Fog-computing-brings-new-business-opportunities-and-disruptions, Blog, TechTarget. Available online: http://internetofthingsagenda.techtarget.com/blog/ (accessed on 15 February 2022).

67. Jalali, F.; Hinton, K.; Ayre, R.; Alpcan, T.; Tucker, R.S. Fog computing may help to save energy in cloud computing. *IEEE J. Sel. Areas Commun.* **2016**, *34*, 1728–1739. [CrossRef]
68. Huerta-Canepa, G.; Lee, D. A virtual cloud computing provider for mobile devices. In *Proceedings of the 1st ACM Workshop on Mobile Cloud Computing & Services: Social Networks and Beyond*; ACM: New York, NY, USA, 2010; pp. 1–5.
69. Jararweh, Y.; Tawalbeh, L.; Ababneh, F.; Dosari, F. Resource efficient mobile computing using cloudlet infrastructure. In Proceedings of the 2013 IEEE Ninth International Conference on Mobile Ad-hoc and Sensor Networks (MSN), Dalian, China, 11–13 December 2013; pp. 373–377.
70. Checko, A.; Christiansen, H.L.; Yan, Y.; Scolari, L.; Kardaras, G.; Berger, M.S.; Dittmann, L. Cloud ran for mobile networks technology overview. *IEEE Commun. Surv. Tutor.* **2015**, *17*, 405–426. [CrossRef]
71. Kliazovich, D.; Granelli, F. Distributed protocol stacks: A framework for balancing interoperability and optimization. In Proceedings of the ICC Workshops-2008 IEEE International Conference on Communications Workshops, Beijing, China, 19–23 July 2008; pp. 241–245.
72. Cheng, J.; Shi, Y.; Bai, B.; Chen, W. Computation offloading in cloud-RAN based mobile cloud computing system. In Proceedings of the 2016 IEEE International Conference on Communications (ICC), Kuala Lumpur, Malaysia, 22–27 May 2016; pp. 1–6.
73. Hung, S.C.; Hsu, H.; Lien, S.Y.; Chen, K.C. Architecture harmonization between cloud radio access networks and fog networks. *IEEE Access* **2015**, *3*, 3019–3034. [CrossRef]
74. Peng, M.; Yan, S.; Zhang, K.; Wang, C. Fog-computing-based radio access networks: Issues and challenges. *IEEE Netw.* **2016**, *30*, 46–53. [CrossRef]
75. Hu, Y.C.; Patel, M.; Sabella, D.; Sprecher, N.; Young, V. Mobile edge computing–a key technology towards 5G. *ETSI White Pap.* **2015**, *11*, 1–16.
76. Giust, F.; Verin, G.; Antevski, K.; Joey, C.; Fang, Y.; Featherstone, W.; Fontes, F.; Frydman, D.; Li, A.; Manzalini, A.; et al. MEC deployments in 4G and evolution towards 5G. *ETSI White Pap.* **2018**, *24*, 1–24.
77. Kadiyala, K.P.; Cobb, J.A. Inter-as traffic engineering with SDN. In Proceedings of the 2017 IEEE Conference on Network Function Virtualization and Software Defined Networks (NFV-SDN), Berlin, Germany, 6–8 November 2017; pp. 1–7.
78. Mirkhanzadeh, B.; Shakeri, A.; Shao, C.; Razo, M.; Tacca, M.; Galimberti, G.M.; Martinelli, G.; Cardani, M.; Fumagalli, A. An SDN-enabled multi-layer protection and restoration mechanism. *Opt. Switch. Netw.* **2018**, *30*, 23–32. [CrossRef]
79. Davies, A. Cisco Pushes IoT Analytics to the Extreme Edge with Mist Computing. Blog, Rethink Research. Available online: http://rethinkresearch.biz/articles/cisco-pushes-iot-analytics-extreme-edge-mist-computing-2 (accessed on 19 December 2014).
80. Preden, J.S.; Tammemäe, K.; Jantsch, A.; Leier, M.; Riid, A.; Calis, E. The benefits of self-awareness and attention in fog and mist computing. *Computer* **2015**, *48*, 37–45. [CrossRef]
81. Silva, P.M.P.; Rodrigues, J.; Silva, J.; Martins, R.; Lopes, L.; Silva, F. Using edge-clouds to reduce load on traditional wifi infrastructures and improve quality of experience. In Proceedings of the 2017 IEEE 1st International Conference on Fog and Edge Computing (ICFEC), Madrid, Spain, 14–15 May 2017; pp. 61–67.
82. Salem, A.; Nadeem, T. Lamen: Leveraging resources on anonymous mobile edge nodes. In *Proceedings of the Eighth Wireless of the Students, by the Students, and for the Students Workshop*; ACM: New York, NY, USA, 2016; pp. 15–17.
83. Morabito, R. Virtualization on internet of things edge devices with container technologies: A performance evaluation. *IEEE Access* **2017**, *5*, 8835–8850. [CrossRef]
84. Abdelwahab, S.; Hamdaoui, B.; Guizani, M.; Znati, T. Cloud of things for sensing-as-a-service: Architecture, algorithms, and use case. *IEEE Internet Things J.* **2016**, *3*, 1099–1112. [CrossRef]
85. Jang, M.; Schwan, K.; Bhardwaj, K.; Gavrilovska, A.; Avasthi, A. Personal clouds: Sharing and integrating networked resources to enhance end user experiences. In Proceedings of the IEEE INFOCOM 2014-IEEE Conference on Computer Communications, Toronto, Canada, 27 April–2 May 2014; pp. 2220–2228.
86. Sathiaseelan, A.; Lertsinsrubtavee, A.; Jagan, A.; Baskaran, P.; Crowcroft, J. Cloudrone: Micro clouds in the sky. In *Proceedings of the 2nd Workshop on Micro Aerial Vehicle Networks, Systems, and Applications for Civilian Use*; ACM: New York, NY, USA, 2016; pp. 41–44.
87. Habak, K.; Zegura, E.W.; Ammar, M.; Harras, K.A. Workload management for dynamic mobile device clusters in edge femtoclouds. In Proceedings of the Second ACM/IEEE Symposium on Edge Computing; ACM: New York, NY, USA, 2017; pp. 1–14.
88. Chang, H.; Hari, A.; Mukherjee, S.; Lakshman, T. Bringing the cloud to the edge. In Proceedings of the 2014 IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS), Toronto, Canada, 27 April–2 May 2014; pp. 346–351.
89. Bhardwaj, K.; Shih, M.-W.; Agarwal, P.; Gavrilovska, A.; Kim, T.; Schwan, K. Fast, scalable and secure onloading of edge functions using airbox. In Proceedings of the 2016 IEEE/ACM Symposium on Edge Computing (SEC), Washington, DC, USA, 27–28 October 2016; pp. 14–27.
90. Villari, M.; Fazio, M.; Dustdar, S.; Rana, O.; Ranjan, R. Osmotic computing: A new paradigm for edge/cloud integration. *IEEE Cloud Comput.* **2016**, *3*, 76–83. [CrossRef]
91. Morshed, A.; Jayaraman, P.P.; Sellis, T.; Georgakopoulos, D.; Villari, M.; Ranjan, R. Deep osmosis: Holistic distributed deep learning in osmotic computing. *IEEE Cloud Comput.* **2018**, *4*, 22–32. [CrossRef]
92. Mosenia, A.; Jha, N.K. A comprehensive study of security of Internet-of-Things. *IEEE Trans. Emerg. Top. Comput.* **2017**, *5*, 586–602. [CrossRef]

93. Yi, S.; Qin, Z.; Li, Q. Security and privacy issues of fog computing: A survey. In Proceedings of the International Conference on Wireless Algorithms, Systems, and Applications, Qufu, China, 10–12 August 2015; pp. 685–695.
94. Walters, J.P.; Liang, Z.; Shi, W.; Chaudhary, V. Wireless sensor network security: A survey. *Secur. Distrib. Grid Mob. Pervasive Comput.* **2007**, *1*, 367–409.
95. Xie, H.; Yan, Z.; Yao, Z.; Atiquzzaman, M. Data collection for security measurement in wireless sensor networks: A survey. *IEEE Internet Things J.* **2018**, *6*, 2205–2224. [CrossRef]
96. Keerthika, M.; Shanmugapriya, D. Wireless sensor networks: Active and passive attacks-vulnerabilities and countermeasures. *Glob. Transit. Proc.* **2021**, *2*, 362–367. [CrossRef]
97. Butun, I.; Österberg, P.; Song, H. Security of the Internet of Things: Vulnerabilities, attacks, and countermeasures. *IEEE Commun. Surv. Tutor.* **2019**, *22*, 616–644. [CrossRef]
98. Lu, Y.; Xu, L.D. Internet of Things (IoT) cybersecurity research: A review of current research topics. *IEEE Internet Things J.* **2019**, *6*, 2103–2115. [CrossRef]
99. Ni, J.; Lin, X.; Shen, X.S. Toward edge-assisted Internet of Things: From security and efficiency perspectives. *IEEE Netw.* **2019**, *33*, 50–57. [CrossRef]
100. Porambage, P.; Okwuibe, J.; Liyanage, M.; Ylianttila, M.; Taleb, T. Survey on multi-access edge computing for Internet of Things realization. *IEEE Commun. Surv. Tuts.* **2018**, *20*, 2961–2991. [CrossRef]
101. Xiao, Y.; Jia, Y.; Liu, C.; Cheng, X.; Yu, J.; Lv, W. Edge computing security: State of the art and challenges. *Proc. IEEE* **2019**, *107*, 1608–1631. [CrossRef]
102. 100 mb Password Dictionary. 2017. Available online: https://github.com/danielmiessler/SecLists/tree/master/Passwords (accessed on 15 February 2022).
103. Hammer-Lahav, E.; Recordon, D.; Hardt, D. The OAuth 1.0 Protocol. In *Document RFC 5849*; IETF: Fremont, CA, USA, 2010.
104. Hardt, D. The OAuth 2.0 Authorization Framework. In *Document RFC 6749*; IETF: Fremont, CA, USA, 2012.
105. Liu, D.; Yan, Z.; Ding, W.; Atiquzzaman, M. A survey on secure data analytics in edge computing. *IEEE Internet Things J.* **2019**, *6*, 4946–4967. [CrossRef]
106. Kolias, C.; Kambourakis, G.; Stavrou, A.; Voas, J. DDoS in the IoT: Mirai and other botnets. *Computer* **2017**, *50*, 80–84. [CrossRef]
107. Yang, Y.; Wu, L.; Yin, G.; Li, L.; Zhao, H. A survey on security and privacy issues in Internet-of-Things. *IEEE Internet Things J.* **2017**, *4*, 1250–1258. [CrossRef]
108. Zhang, J.; Chen, B.; Zhao, Y.; Cheng, X.; Hu, F. Data security and privacy-preserving in edge computing paradigm: Survey and open issues. *IEEE Access* **2018**, *6*, 18209–18237. [CrossRef]
109. He, D.; Chan, S.; Guizani, M. Security in the Internet of Things supported by mobile edge computing. *IEEE Commun. Mag.* **2018**, *56*, 56–61. [CrossRef]
110. Omoniwa, B.; Hussain, R.; Javed, M.A.; Bouk, S.H.; Malik, S.A. Fog/edge computing-based IoT (FECIoT): Architecture, applications, and research issues. *IEEE Internet Things J.* **2019**, *6*, 4118–4149. [CrossRef]
111. Mukherjee, M.; Matam, R.; Shu, L.; Maglaras, L.; Ferrag, M.A.; Choudhury, N.; Kumar, V. Security and privacy in fog computing: Challenges. *IEEE Access* **2017**, *5*, 19293–19304. [CrossRef]
112. Karakehayov, Z. Using REWARD to detect team black-hole attacks in wireless sensor networks. *Proc. Workshop Real-World Wirel. Sens. Netw.* **2005**, *20*, 20–21.
113. Revathi, B.; Geetha, D. A survey of cooperative black and gray hole attack in MANET. *Int. J. Comput. Sci. Manage. Res.* **2012**, *1*, 205–208.
114. Garcia-Morchon, O.; Kumar, S.; Struik, R.; Keoh, S.; Hummen, R. Security Considerations in the IP-Based Internet of Things. Available online: https://tools.ietf.org/html/draft-garcia-core-security-04 (accessed on 1 February 2016).
115. Wallgren, L.; Raza, S.; Voigt, T. Routing attacks and countermeasures in the RPL-based Internet of Things. *Int. J. Distrib. Sens. Netw.* **2013**, *9*, 794326. [CrossRef]
116. Singh, P.V.; Jain, S.; Singhai, J. Hello flood attack and its countermeasures in wireless sensor networks. *Int. J. Comput. Sci.* **2010**, *7*, 23.
117. Douceur, J.R. The Sybil attack. In *Peer-to-Peer Systems*; Springer: London, UK, 2002; pp. 251–260.
118. Chen, S.; Jiang, Y.; Wen, H.; Liu, W.; Chen, J.; Lei, W.; Xu, A. A novel terminal security access method based on edge computing for IoT. In Proceedings of the 2018 International Conference on Networking and Network Applications (NaNA), Xi'an, China, 15 October 2018; pp. 394–398.
119. Hong, K.; Lillethun, D.; Ramachandran, U.; Ottenwälder, B.; Koldehofe, B. Mobile fog: A programming model for large-scale applications on the Internet of Things. In Proceedings of the second ACM SIGCOMM workshop on Mobile cloud computing, Hong Kong, China, 16 August 2013; pp. 15–20.
120. Cisco Press, Developing Network Security Strategies. 2010. Available online: https://www.ciscopress.com/articles/article.asp?p=1626588&seqNum=2 (accessed on 4 October 2010).
121. Yu, W.; Liang, F.; He, X.; Hatcher, W.G.; Lu, C.; Lin, J.; Yang, X. A survey on the edge computing for the Internet of Things. *IEEE Access* **2018**, *6*, 6900–6919. [CrossRef]
122. Wang, T.; Zhang, G.; Liu, A.; Bhuiyan, M.Z.A.; Jin, Q. A secure IoT service architecture with an efficient balance dynamics based on cloud and edge computing. *IEEE Internet Things J.* **2019**, *6*, 4831–4843. [CrossRef]
123. Lin, F.; Zhou, Y.; An, X.; You, I.; Choo, K.R. Fair resource allocation in an intrusion-detection system for edge computing: Ensuring the security of Internet of Things devices. *IEEE Consum. Electron. Mag.* **2018**, *7*, 45–50. [CrossRef]

124. Chekired, D.A.; Khoukhi, L.; Mouftah, H.T. Fog-based distributed intrusion detection system against false metering attacks in smart grid. In Proceedings of the ICC 2019–2019 IEEE International Conference on Communications (ICC), Shanghai, China, 20–24 May 2019; pp. 1–6.
125. Singh, T.; Aksanli, B. Real-time traffic monitoring and SQL injection attack detection for edge networks. In Proceedings of the 15th ACM International Symposium on QoS and Security for Wireless and Mobile Networks, Miami Beach, FL, USA, 25 November 2019; pp. 29–36.
126. Esiner, E.; Datta, A. Layered security for storage at the edge: On decentralized multi-factor access control. In Proceedings of the 17th International Conference on Distributed Computing and Networking, Singapore, 4–7 January 2016; p. 9.
127. Cui, H.; Yi, X.; Nepal, S. Achieving scalable access control over encrypted data for edge computing networks. *IEEE Access* **2018**, *6*, 30049–30059. [CrossRef]
128. Liang, K.; Au, M.H.; Liu, J.K.; Susilo, W.; Wong, D.S.; Yang, G.; Yu, Y.; Yang, A. A secure and efficient cipher text-policy attribute-based proxy re-encryption for cloud data sharing. *Future Gener. Comput. Syst.* **2015**, *52*, 95–108. [CrossRef]
129. Mollah, M.B.; Azad, M.A.K.; Vasilakos, A. Secure data sharing and searching at the edge of cloud-assisted Internet of Things. *IEEE Cloud Comput.* **2017**, *4*, 34–42. [CrossRef]
130. Siegel, J.E.; Kumar, S.; Sarma, S.E. The future Internet of Things: Secure, efficient, and model-based. *IEEE Internet Things J.* **2018**, *5*, 2386–2398. [CrossRef]
131. Ibrahim, M.H. Octopus: An edge-fog mutual authentication scheme. *Int. J. Netw. Secur.* **2016**, *18*, 1089–1101.
132. Mouratidis, H.; Giorgini, P. Security attack testing (SAT): Testing the security of information systems at design time. *Inf. Syst.* **2007**, *32*, 1166–1183. [CrossRef]
133. Du, M.; Wang, K.; Chen, Y.; Wang, X.; Sun, Y. Big data privacy preserving in multi-access edge computing for heterogeneous Internet of Things. *IEEE Commun. Mag.* **2018**, *56*, 62–67. [CrossRef]
134. Wei, W.; Xu, F.; Li, Q. Mobishare: Flexible privacy-preserving location sharing in mobile online social networks. In Proceedings of the 2012 IEEE INFOCOM, Orlando, FL, USA, 25–30 March 2012; pp. 2616–2620.
135. Jan, M.A.; Zhang, W.; Usman, M.; Tan, Z.; Khan, F.; Luo, E. Smartedge: An end-to-end encryption framework for an edge-enabled smart city application. *J. Netw. Comput. Appl.* **2019**, *137*, 1–10. [CrossRef]
136. Xu, R.; Chen, Y.; Blasch, E.; Chen, G. BlendCAC: A blockchain-enabled decentralized capability-based access control for IoTs. In Proceedings of the 2018 IEEE International Conference on Internet of Things (iThings) and IEEE Green Computing and Communications (GreenCom) and IEEE Cyber, Physical and Social Computing (CPSCom) and IEEE Smart Data (SmartData), Halifax, NS, Canada, 30 July–3 August 2018; pp. 1027–1034.
137. Kang, J.; Yu, R.; Huang, X.; Wu, M.; Maharjan, S.; Xie, S.; Zhang, Y. Blockchain for secure and efficient data sharing in vehicular edge computing and networks. *IEEE Internet Things J.* **2019**, *6*, 4660–4670. [CrossRef]
138. Gai, K.; Wu, Y.; Zhu, L.; Xu, L.; Zhang, Y. Permissioned blockchain and edge computing empowered privacy-preserving smart grid networks. *IEEE Internet Things J.* **2019**, *6*, 7992–8004. [CrossRef]
139. Ali, A.; Shah, G.A.; Farooq, M.O.; Ghani, U. Technologies and challenges in developing machine-to-machine applications: A survey. *J. Netw. Comput. Appl.* **2017**, *83*, 124–139. [CrossRef]
140. Muller, N.J. *Networking A to Z*; McGraw-Hill Professional Publishing: New York, NY, USA, 2002; pp. 45–47.
141. Roberts, C.M. Radio frequency identification (RFID). *Comput. Secur.* **2006**, *25*, 18–26. [CrossRef]
142. Chen, M.; Miao, Y.; Hao, Y.; Hwang, K. NarrowBand Internet of Things (NB-IoT). *IEEE Access* **2017**, *5*, 20557–20577. [CrossRef]
143. Grant, S. *3GPP Low Power Wide Area Technologies-GSMA (White Paper)*; GSM Association: London, UK, 2016; p. 49.
144. Standardization of NB-IOT Completed. 3GPP: 2016, p. 1. Available online: https://www.3gpp.org/news-events/1785-nb_iot_complete (accessed on 22 June 2016).
145. Huurdeman, A.A. *The Worldwide History of Telecommunications*; John Wiley & Sons: Hoboken, NJ, USA, 2003; p. 529.
146. Gohil, A.; Modi, H.; Patel, S.K. 5G technology of mobile communication: A survey. In Proceedings of the 2013 International Conference on Intelligent Systems and Signal Processing (ISSP), Vallabh Vidyanagar, India, 1–2 March 2013; pp. 288–292.
147. Hoffman, C. What is 5G, and how fast will it be? In How-To Geek Website, How-To Geek LLC. 2019. Available online: https://www.howtogeek.com/340002/what-is-5g-and-how-fast-will-it-be/ (accessed on 15 February 2022).
148. Postcapes, IoT Standards & Protocols Guide—2019 Comparisons on Network, Wireless Comms, Security, Industrial. 2019. Available online: https://www.postscapes.com/internet-of-things-protocols/ (accessed on 20 January 2019).
149. Benefits of Mobile Ticketing in Public Transport. 2019. Available online: https://www.discoverpassenger.com/2017/01/23/benefits-of-mobile-ticketing/ (accessed on 23 January 2017).
150. Juntunen, A.; Luukkainen, S.; Tuunainen, V.K. Deploying NFC technology for mobile ticketing services—Identifcation of critical business model issues. In Proceedings of the 2010 Ninth International Conference on Mobile Business and 2010 Ninth Global Mobility Roundtable (ICMB-GMR), Athens, Greece, 17–19 June 2010; pp. 82–90.
151. Ghanem, B.; Schneider, J.; Shalaby, M.; Elnily, U. System and Method for Crowd Counting and Tracking. U.S. Patent 9,361,524, 7 June 2016.
152. Choudhury, T.; Borriello, G.; Consolvo, S.; Haehnel, D.; Harrison, B.; Hemingway, B.; Hightower, J.; Pedja, P.; Koscher, K.; LaMarca, A.; et al. The mobile sensing platform: An embedded activity recognition system. *IEEE Pervasive Comput.* **2008**, *7*, 32–41. [CrossRef]

153. Lester, J.; Choudhury, T.; Borriello, G. A practical approach to recognizing physical activities. In *International Conference on Pervasive Computing*; Springer: Berlin/Heidelberg, Germany, 2006; Volume 3968, pp. 1–16.
154. Coskun, D.; Incel, O.D.; Ozgovde, A. Phone position/placement detection using accelerometer: Impact on activity recognition. In Proceedings of the 2015 IEEE Tenth International Conference on Intelligent Sensors, Sensor Networks and Information Processing (ISSNIP), Singapore, 7–9 April 2015; pp. 1–6.
155. Fang, S.H.; Fei, Y.X.; Xu, Z.; Tsao, Y. Learning transportation modes from smartphone sensors based on deep neural network. *IEEE Sens. J.* **2017**, *17*, 6111–6118. [CrossRef]
156. Hemminki, S.; Nurmi, P.; Tarkoma, S. Accelerometer-based transportation mode detection on smartphones. In Proceedings of the 11th ACM Conference on Embedded Networked Sensor Systems—SenSys'13, Roma, Italy, 11–15 November 2013; ACM Press: New York, NY, USA, 2013; pp. 1–14.
157. Stenneth, L.; Wolfson, O.; Yu, P.S.; Xu, B. Transportation mode detection using mobile phones and GIS information. In Proceedings of the 19th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems—GIS '11, Chicago, IL, USA, 1–4 November 2011; ACM Press: New York, NY, USA, 2011; pp. 54–63.
158. Ho, B.J.; Martin, P.; Swaminathan, P.; Srivastava, M. From pressure to path: Barometer-based vehicle tracking. In Proceedings of the 2nd ACM International Conference on Embedded Systems for Energy-Efficient Built Environments—BuildSys '15, Seoul, Korea, 4–5 November 2015; ACM Press: New York, NY, USA, 2015; pp. 65–74.
159. Nawaz, S.; Mascolo, C. Mining users' significant driving routes with low-power sensors. In Proceedings of the 12th ACM Conference on Embedded Network Sensor Systems—SenSys '14, Memphis, TEN, USA, 3–6 November 2014; ACM Press: New York, NY, USA, 2014; pp. 236–250.
160. Sankaran, K.; Zhu, M.; Guo, X.F.; Ananda, A.L.; Chan, M.C.; Peh, L.S. Using mobile phone barometer for low-power transportation context detection. In Proceedings of the 12th ACM Conference on Embedded Network Sensor Systems—SenSys '14, Memphis, TEN, USA, 3–6 November 2014; ACM Press: New York, NY, USA, 2014; pp. 191–205.
161. Vanini, S.; Faraci, F.; Ferrari, A.; Giordano, S. Using barometric pressure data to recognize vertical displacement activities on smartphones. *Comput. Commun.* **2016**, *87*, 37–48. [CrossRef]
162. Won, M.; Mishra, A.; Son, S.H. Hybridbaro: Mining driving routes using barometer sensor of smartphone. *IEEE Sens. J.* **2017**, *17*, 6397–6408. [CrossRef]
163. Won, M.; Zhang, S.; Chekuri, A.; Son, S.H. Enabling energy-efficient driving route detection using built-in smartphone barometer sensor. In Proceedings of the 2016 IEEE 19th International Conference on Intelligent Transportation Systems (ITSC), Rio de Janeiro, Brazil, 1–4 November 2016; pp. 2378–2385.
164. Gunady, S.; Keoh, S.L. A non-GPS Based location tracking of public buses using bluetooth proximity beacons. In Proceedings of the 2019 IEEE 5th World Forum on Internet of Things (WF-IoT), Limerick, Ireland, 15–18 April 2019; pp. 606–611.
165. Narzt, W.; Mayerhofer, S.; Weichselbaum, O.; Haselböck, S.; Höfler, N. Bluetooth low energy as enabling technology for Be-In/Be-Out systems. In Proceedings of the 2016 13th IEEE Annual Consumer Communications Networking Conference (CCNC), Las Vegas, NV, USA, 9–12 January 2016; pp. 423–428.
166. Sarkar, C.; Treurniet, J.J.; Narayana, S.; Prasad, R.V.; de Boer, W. SEAT: Secure energy-efficient automated public transport ticketing system. *IEEE Trans. Green Commun. Netw.* **2018**, *2*, 222–233. [CrossRef]
167. Tuveri, G.; Garau, M.; Sottile, E.; Pintor, L.; Gravellu, M.; Atzori, L.; Meloni, I. Automating ticket validation: A key strategy for fare clearing and service planning. In Proceedings of the 2019 6th International Conference on Models and Technologies for Intelligent Transportation Systems (MT-ITS), Kraków, Poland, 5–7 June 2019; pp. 1–10.
168. Wieczorek, B.; Poniszewska-Maranńda, A. Be in/Be out model for intelligent transport in SmartCity approach. In Proceedings of the 17th International Conference on Advances in Mobile Computing & Multimedia (MoMM2019); Association for Computing Machinery: New York, NY, USA, 2019; pp. 226–230.
169. Jain, V.; Khurana, Y.; Kharbanda, M.; Mehta, K. Be a ticket-beacon based ticketing system. *Recent Pat. Comput. Sci.* **2019**, *12*, 611–619.
170. Yu, M.C.; Yu, T.; Wang, S.C.; Lin, C.J.; Chang, E.Y. Big data small footprint: The design of a low-power classifier for detecting transportation modes. *Proc. VLDB Endow.* **2014**, *7*, 1429–1440. [CrossRef]
171. Hoseini-Tabatabaei, S.A.; Gluhak, A.; Tafazolli, R. A survey on smartphone-based systems for opportunistic user context recognition. *ACM Computing Surv. CSUR* **2013**, *45*, 1–51. [CrossRef]
172. Alam, T. Middleware implementation in cloud-MANET mobility model for internet of smart devices. *Arxiv Prepr.* **2019**, arXiv:1902.09744. [CrossRef]
173. Alam, T.; Benaida, M. CICS: Cloud–internet communication security framework for the internet of smart devices. *Int. J. Interact. Mob. Technol. Ijim* **2018**, *12*. [CrossRef]
174. Ronen, E.; Flynn, C.O.; Shamir, A.; Weingarten, A.O. IoT Goes nuclear: Creating a ZigBee chain reaction. In Proceedings of the 2017 IEEE Symposium on Security and Privacy (SP), San Jose, CA, USA, 22–24 May 2017; pp. 195–212.
175. Kifer, D.; Machanavajjhala, A. No free lunch in data privacy. In *Proceedings of the 2011 ACM SIGMOD International Conference on Management of Data*; ACM: New York, NY, USA, 2011; pp. 193–204.
176. Haeberlen, A.; Pierce, B.C.; Narayan, A. Differential privacy under fire. In Proceedings of the 20th USENIX Security Symposium (USENIX Security 11), San Francisco, CA, USA, 8–12 August 2011; USENIX Association: Berkeley, CA, USA, 2011; p. 33.

177. Lee, S.; Shih, M.W.; Gera, P.; Kim, T.; Kim, H.; Peinado, M. Inferring fine-grained control flow inside SGX enclaves with branch shadowing. In Proceedings of the 26th USENIX Security Symposium (USENIX Security 17), Vancouver, BC, Canada, 16–18 August 2017; USENIX Association: Berkeley, CA, USA, 2017; pp. 557–574.
178. Felt, A.P.; Chin, E.; Hanna, S.; Song, D.; Wagner, D. Android permissions demystified. In Proceedings of the 18th ACM conference on Computer and Communications Security, New York, NY, USA, 17–21 October 2011; pp. 627–638.
179. Tian, Y.; Zhang, N.; Lin, Y.H.; Wang, X.; Ur, B.; Guo, X.; Tague, P. SmartAuth: User-centered authorization for the Internet of Things. In Proceedings of the 26th USENIX Security Symposium (USENIX Security 17), Vancouver, BC, Canada, 16–18 August 2017; USENIX Association: Berkeley, CA, USA, 2017; pp. 361–378.