Akman, Gizem; Ginzboorg, Philip; Niemi, Valtteri

Privacy-Aware Access Protocols for MEC Applications in 5G

Published in:
Network

DOI:
10.3390/network2020014

Published: 01/04/2022

Document Version
Publisher's PDF, also known as Version of record

network

Article

Privacy-Aware Access Protocols for MEC Applications in 5G †

Gizem Akman [1,2,*], Philip Ginzboorg [3,4] and Valtteri Niemi [1,2]

[1] Department of Computer Science, University of Helsinki, Pietari Kalmin Katu 5, 00560 Helsinki, Finland; valtteri.niemi@helsinki.fi
[2] Helsinki Institute for Information Technology (HIIT), Konemiehentie 2, 02150 Espoo, Finland
[3] Huawei Technologies Oy, Itämerenkatu 9, 00180 Helsinki, Finland; philip.ginzboorg@huawei.com
[4] Department of Communications and Networking, Aalto University, Maarintie 8, 02150 Espoo, Finland
* Correspondence: gizem.akman@helsinki.fi
† This paper is an extended version of our paper published in Computational Science and Its Applications (ICCSA 2021)-International Workshop on Privacy in the Cloud/Edge/IoT World (PCEIoT 2021).

Abstract: Multi-access edge computing (MEC) is one of the emerging key technologies in fifth generation (5G) mobile networks, providing reduced end-to-end latency for applications and reduced load in the transport network. This paper proposes mechanisms to enhance user privacy in MEC within 5G. We consider a basic MEC usage scenario, where the user accesses an application hosted in the MEC platform via the radio access network of the mobile network operator (MNO). First, we create a system model based on this scenario. Second, we define the adversary model and give the list of privacy requirements for this system model. We also analyze the impact on user privacy when some of the parties in our model share information that is not strictly needed for providing the service. Third, we introduce a privacy-aware access protocol for the system model and analyze this protocol against the privacy requirements.

Keywords: MEC; 5G; IoT; identity privacy; data unlinkability; data confidentiality

Citation: Akman, G.; Ginzboorg, P.; Niemi, V. Privacy-Aware Access Protocols for MEC Applications in 5G. *Network* **2022**, *2*, 203–224. https://doi.org/10.3390/network2020014

Academic Editors: Michele Mastroianni, Francesco Palmieri and Youn-Hee Han

Received: 30 December 2021
Accepted: 30 March 2022
Published: 1 April 2022

1. Introduction

The standardization of mobile edge computing (MEC) was initiated by the European Telecommunication Standards Institute (ETSI) in 2014. The purpose was to bring cloud computing to the mobile networks closer to the mobile users [1,2]. In 2017, ETSI changed the term Mobile Edge Computing to multi-access edge computing while keeping the acronym MEC [3]. The 3rd Generation Partnership Project (3GPP) describes edge computing as an enabler that allows the hosting of services, provided by mobile network operators or by third parties, near the mobile user [4]. The local hosting of services has the benefits of lower end-to-end latency and reduced load in the core transportation network [5].

Cloud computing is a requisite for IoT devices to manage their data, and MEC provides cloud computing resources at the edge of the network. With the increment of IoT devices, the amount of produced data is increasing along with the need for storage and computing powers [6]. Therefore, computational offloading is emerging so that these IoT devices can take advantage of performing tasks locally in MEC rather than relying on cloud servers [7]. In addition to reducing the processing time and power, it helps protect user data privacy since transferring the local data to the cloud is a threat to the privacy of the data. For example, with a deep learning framework, data can be processed in MEC, and only the output is sent to the cloud server instead of the original dataset [8]. Thus, MEC can provide efficient, reliable, and secure support to IoT networks, which would also enable improvements of AI, blockchain, and big data techniques [9]. In addition, vertical industries (e.g., smart city services and vehicle-to-everything (V2X) [10]), distributed content delivery and caching are some examples of services that would benefit from the MEC applications [11].

Fifth-generation (5G) mobile networks have been designed to provide high bandwidth, low latency, and low operational costs [12]. One other aim of the development of 5G is to provide flexible deployment of the data plane [2]. The flexibility and the advances of 5G would help to support MEC and the deployment of innovative applications and intelligent services which IoT devices provide [9]. MEC is considered as an application function (AF) in the 5G system, and it is connected to the 5G network through the user plane function (UPF) [13]. While MEC is regarded as a key technology of 5G, MEC is already used in existing 4G networks [2].

A schematic illustration of MEC deployment in 5G networks is shown in Figure 1. A client of an application in a mobile device can be served not only by the main server of the application over the internet, but also by a local MEC application server in the network of the operator. To enable the latter, a new entity that is built upon data center technologies and called MEC host is deployed near the edge of the mobile network. More specifically, it can be deployed to several locations that are within or beside the radio access network (RAN) [3]. Either the mobile network operator (MNO) or third-party cloud services providers can deploy the MEC host to the mobile network [4]. Amazon Web Services, Microsoft Azure, Google Cloud, IBM Cloud, and Oracle Cloud are examples of third-party cloud service providers [14,15].

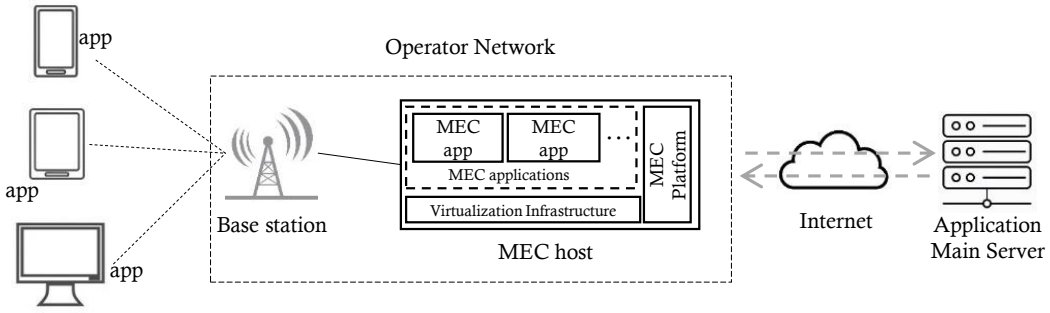

Figure 1. Overview of the network after deployment of MEC based on [13,16].

MEC usage involves a large amount of data passing between the user at the edge of the mobile network and the MEC application in the MEC host. Parts of these data may be personal and sensitive, which raises privacy concerns. The network entities involved in MEC usage could belong to the same company or different companies. Independently of who owns the network entities, analyzing the MEC-related data could be helpful. We will come back to these privacy issues in Section 3.2.1.

Prior works on MEC security and privacy focus on protecting user privacy against outsiders. Our first contribution is to identify privacy problems in the situation where honest-but-curious parties (in our case, MNO, MEC, and MEC applications) work together to provide services to customers. Our second contribution is a set of privacy requirements for this situation. As a third contribution, we introduce a solution and analyze how it meets the privacy requirements.

Our research follows the design science methodology [17], which has six steps: problem identification and motivation, definition of the objectives for a solution, design and development, demonstration, evaluation, and communication [18].

The paper is structured as follows. Review of relevant information for the paper and related works are explained in Section 2. We continue by presenting the statement of the problem in Section 3. In this section, we introduce the system model in Section 3.1, we define the adversary model in Section 3.2, and we list the requirements that will help to protect the privacy of the user in Section 3.3. Then, Section 4 is for the results and discussions. Considering the adversary model and privacy requirements, we present a solution for privacy-preserving access to MEC applications in Section 4.1. In Section 4.2, we explain how our solution meets the privacy requirements. We finalize our paper with final remarks, which include the conclusion and the future work in Section 5.

2. Review and Related Work

In this section, we first explain the relevant concepts for our study, and then we present related work.

2.1. Review

We focus on three notions related to privacy: data confidentiality, identity confidentiality, and unlinkability.

Confidentiality is "*the act of preventing unauthorized entities from reading or accessing sensitive materials*" [19]. In our paper, we focus on data and identity confidentiality. Data confidentiality means "*protecting data against unintentional, unlawful, or unauthorized access, disclosure, or theft*" [20]. Only the entities who require these data to provide services should have access to the data.

Identity confidentiality can be defined as follows: "*the permanent identity of a user to whom a service is delivered cannot be eavesdropped on the radio access link*" [21]. The user might have various identifiers for MNO and MEC application. International Mobile Subscriber Identity (IMSI) is the permanent identifier of the subscriber in a mobile network. The IMSI is sent over the radio access link as rarely as possible in legacy mobile networks, and a randomly generated temporary identifier is sent instead. This randomly generated identifier is called the temporary mobile subscriber identity (TMSI) in 2G and 3G systems, and the globally unique temporary identifier (GUTI) in the 4G system. In the 5G system, the temporary user identity is called 5G-GUTI, and the permanent user identity is called the subscription permanent identifier (SUPI). For simplicity, we use the terms IMSI and TMSI to designate the permanent and the temporary identities of the mobile user throughout the rest of this paper, even when the discussion is about 5G. The identifiers should not be learned by parties who do not need them to provide services.

The definition of unlinkability is given in [22]: "*unlinkability of two or more items of interest from an attacker's perspective means that within the system, the attacker cannot sufficiently distinguish whether these items are related or not*". We require the unlinkability of both messages and user identifiers.

The following concepts are relevant to our work: virtual private network (VPN) and onion routing (OR). VPN helps to establish a secure private channel between two entities in a public network instead of building a private network by using the tunneling protocol [23]. The tunneling protocol enables a safe way of transport for network services that are not supported directly by the elemental network [24]. This way, data can be sent and received securely by cost-efficient and reliable methods in the public network. In addition to providing a secure connection, VPN helps to anonymize the traffic of the user on the internet. It also makes it harder for attackers to track the activities of the user over the internet [25].

Onion routing constructs a layered object to direct an anonymous, bi-directional, real-time virtual circuit between two entities (sender and receiver) in the network [26]. In between the two communicating parties, there are several nodes, which are called onion routers. A default route exists of three routers: entry router (sender), intermediate router, and exit router (receiver) [27]. The number of intermediate routers can vary depending on the requirements of the route. The onion proxy knows the onion routers in the network and holds the information, such as IP addresses, public keys, and bandwidth, of these routers [27]. Then, the proxy determines the intermediate routers of the virtual path. The sender encrypts the messages layer by layer with the public keys of each router and sends the message. Each router decrypts its own layer and forwards the message to the next router. The receiver decrypts the last layer and reads the plaintext [28]. It should be noted that each router knows only the previous and next routers of the path, and the rest of the path stays anonymous [27,28].

2.2. Related Work

The surveys by Ranaweera et al. [19,29] present background information on MEC and examine the security and privacy vulnerabilities of MEC deployment scenarios in 5G use cases. The security solutions and countermeasures for these use cases are also provided. Du et al. [30] describe the privacy issues in MEC and introduce a case study considering machine learning privacy-preserving for MEC. Okwuibe et al. [31] discuss the security of cloud computing and MEC in 5G. Privacy and security threats for different use cases are defined, and decent solutions are proposed for these threats. Kim et al. and Ranaweera et al. [32,33] emphasize that the MEC system hosts heterogeneous third-party applications that are based on cloud and virtualization technologies, which can cause security threats. Kim et al. [32] present the structure and the features of MEC frameworks and security threats in the 5G system. Ranaweera et al. [33] analyze the MEC-related threats in 5G and propose how to mitigate them.

He et al. [34] introduce the privacy issues due to the location of MEC-enabled IoT devices and present a learning algorithm to mitigate this privacy issue. Lee et al. [35] also raise the problems due to location privacy. They propose a protocol to improve user anonymity and allow traceability for the trusted parties for providing services.

Zhang et al. [36] raise privacy concerns related to cellular networks and introduce a solution using MEC that provides better privacy protection compared to typical VPN and cellular networks.

Kotulski et al. [37] introduce a new access control architecture for the 5G MEC network to prevent non-authorized access to edge services and to protect network and computation resources from unnecessary allocations.

Khan et al., Ahmad et al., and Carvalho et al. [38–40] deal with innovative technologies of 5G, including software-defined networking (SDN), network function virtualization (NFV), and MEC, along with security and privacy concerns when using these technologies. The survey by Khan et al. [38] explores the 5G security model and threat analysis in 5G networks. Ahmad et al. [39] describe the security challenges, possible security solutions, and security standardization activities for 5G systems. The paper by Carvalho et al. [40] proposes and analyzes a risk-aware edge server orchestrator mechanism.

Our work differs from the above prior work in the way that we consider the situation where several parties work together to provide a service to customers while at the same time trying to gather more data about these customers than what is needed to provide their part of the service. For example, these additional data could be the identities and usage patterns relevant only for other parties.

3. Statement of the Problem

In order to formalize the problem of privacy-preserving MEC usage, we introduce a system model, an adversary model, and then identify privacy requirements for the parties in the system model.

3.1. System Model

We base our system model on the scenario where a user wants to use an application APPIFY in a MEC host deployed in a network of a mobile network operator, MNO. The user does not want to reveal the contents of messages to any party other than APPIFY. For simplicity, the user is referred to as Alice in this paper, and we use quotation marks, "Alice" to point to the actual name of the user. The MEC host is also called MEC, and the application in MEC is referred to as APPIFY.

We build an abstract model with four parties: Alice, MNO, MEC, and APPIFY. In order to use APPIFY, Alice needs to send and receive messages over the networks of MNO and MEC. The visualization of the model is presented in Figure 2.

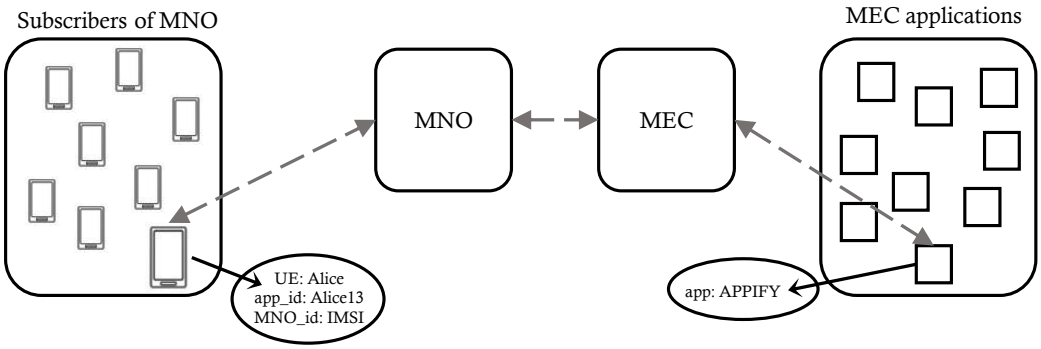

Figure 2. Elements of the system model.

Alice is one subscriber of MNO, among other subscribers. The identifier of Alice for MNO is IMSI. Due to the subscription, MNO may have learned further details about Alice, such as her home address, real name, social security number, and email address.

MEC is deployed in a mobile network of an MNO, and it hosts several applications, including APPIFY. The messages between MNO and APPIFY are sent through MEC. Alice and MEC do not share any identifiers or credentials in our model.

APPIFY is hosted in MEC along with other MEC applications. Typically, the data in the MEC application residing in the MEC host are synchronized regularly with the main server of the application (see Figure 1) to provide up-to-date service to users of the application [41]. Therefore, APPIFY is also in contact with the main server of APPIFY. However, we do not include the main server of APPIFY in our model for simplicity.

The traffic between adjacent parties in the model, Alice-to-MNO, MNO-to-MEC, and MEC-to-APPIFY, is assumed to be secured by separate mechanisms. Therefore, the outsiders, including the other subscribers of MNO and other MEC applications in MEC, cannot learn the content of the messages. However, if there are no additional security mechanisms, MNO and MEC can see the messages between Alice and APPIFY, so they would learn which application she is using, her identity for APPIFY, and the content of the messages. Still, Alice does not want MNO to retain such information about her, and she also wants to stay anonymous toward MEC.

3.2. Adversary Model

We now define types of adversaries that are relevant to the system model described in Section 3.1. These adversary types are Dolev–Yao and honest-but-curious. The choice of these adversary types is justified in Section 3.2.1.

A Dolev–Yao adversary can listen to the traffic on the network and interact with the other entities of the network [42]. This adversary has the ability to "*overhear, intercept, and synthesize any message and is only limited by the constraints of the cryptographic methods used*" [43]. Nevertheless, the Dolev–Yao adversary cannot crack encrypted messages but can only decrypt them if this adversary has the right key [44].

An honest-but-curious adversary is defined as "*a legitimate participant in a communication protocol who will not deviate from the defined protocol but will attempt to learn all possible information from legitimately received messages*" [22]. The local information, e.g., messages in the network, is used to deduce private data without interfering with the traffic [45].

The parties in our system model, MNO, MEC, and APPIFY, are assumed to be honest-but-curious adversaries. They follow the protocol, but still, they are trying to learn as much as possible about the user. Moreover, apart from being honest but curious, MNO, MEC, and APPIFY are passive Dolev–Yao adversaries: they may see all messages sent by other parties, but they do not actively interfere with these messages.

Elements that are not visible in our system model may be Dolev–Yao adversaries. They can see, delete, add, replay, modify, reroute, and reorder the messages [46] since they do not need to be honest. MNO subscribers other than Alice may be malicious, having Dolev–Yao potential. We assume that MEC applications other than APPIFY are honest-but-curious,

passive Dolev–Yao adversaries. All these three types of entities are included in the category of outsiders.

3.2.1. Justification of the Adversary Model

The parties in the system model, MNO, MEC, and APPIFY, are assumed to be honest-but-curious and passive Dolev–Yao adversaries.

We assume that these parties are honest because they conduct business and care about their reputation. If they abuse the authority they have, for example, by selling the customer data to third parties, then they can obtain a bad reputation and lose the trust of the customers; an example case can be found in [47]. In consequence, they would likely comply with the regulations.

A malicious actor may succeed in infiltrating one of these parties, e.g., one of the MNO employees is malicious. This is called an insider attack. We assume that each party has separate mechanisms to detect such attacks and procedures to prevent them, or at least for recovering fast. Therefore, we do not include attacks by malicious insiders in our adversary model.

The parties can be curious about the customers to whom they provide services while still being honest. It is possible to learn the habits of the customers by analyzing their messages, e.g., which services they prefer, how often and when they use these services, and what, in general, is vital to customers. This analysis provides companies with an understanding of what customers want based on their previous behaviors [48]. In the system model, MNO and MEC transfer messages between Alice and APPIFY. MNO and MEC can retrieve these messages for further analysis. In addition, APPIFY might try to learn more about Alice than what she shares with the application.

We also assume that MNO, MEC, and APPIFY are passive Dolev–Yao adversaries. Both MEC and APPIFY have an opportunity to see the messages of Alice when they are sent between Alice and MNO because radio communication is used. It is not as easy to arrange for APPIFY to receive the messages between MNO and MEC or for MNO to receive the messages between MEC and APPIFY. Still, we do not want to rely on the technical difficulties of such passive eavesdropping. However, all three parties are honest, so they do not interfere with the communication by rerouting, deleting, editing, or replaying the messages. Therefore, they are passive, not active, Dolev–Yao adversaries.

We have three different types of entities that are all called outsiders. The first type contains elements that are not even visible in our system model. However, such elements can observe and interfere with the messages sent between the parties. Therefore, these elements can be viewed as Dolev–Yao adversaries. The second type consists of MNO subscribers other than Alice. They might have malicious intentions and possess Dolev–Yao capabilities. It is straightforward to obtain a subscription or buy a prepaid card from MNO, and generally, such fresh subscribers do not have any reputation to lose. These other subscribers have legitimate access to MEC applications, which helps them perform various active attacks, e.g., denial of service (DoS). The third type of outsiders contains MEC applications in the MEC host, other than APPIFY. These applications are in a similar position as APPIFY in the system model. For similar reasons as explained above for the case of APPIFY, it is natural to assume that they are honest-but-curious adversaries with passive Dolev–Yao capabilities.

Alice is not an adversary in our system model because the assets we try to protect are her identities and data.

3.3. Privacy Requirements

The system model in Section 3.1 includes the user Alice and outsiders, and on the other hand, MNO, MEC, and APPIFY, which together serve the user. The data seen by MNO, MEC, and APPIFY include the necessary information to provide the service to Alice. Outsiders do not participate in the protocol.

The security properties of data confidentiality, identity confidentiality, and unlinkability were taken into account when defining privacy requirements for the serving parties: MNO, MEC, and APPIFY, and the outsiders.

Below, we labeled by "D" the requirements related to data confidentiality. These requirements aim to prevent the party from accessing information that it does not need to know.

The requirements related to identity confidentiality are labeled by "I" below. Alice may have several identities for different parties in the system model. The goal of these requirements is to prevent the discovery of identities of the user by parties that do not need them. One party should not know the identities that Alice uses with other parties.

Below, we have labeled by "U" the requirements related to unlinkability. The goal of these requirements is to prevent distinguishing whether two messages belong to the same user, same destination, or include the same identifier.

The list of privacy requirements is given next.

R1-MNO-D: MNO should not learn which MEC application Alice is using.
R2-MNO-D: MNO should not learn the content of what Alice sends to and receives from APPIFY.
R3-MNO-I: MNO should not learn that Alice13 is the identifier of Alice for APPIFY.
R4-MNO-U: MNO should not be able to distinguish whether two messages are related to the same MEC application.
R5-MNO-U: MNO should not be able to distinguish whether two messages are related to the same user identifier for the MEC application.
R6-MEC-D: MEC should not learn the content that Alice sends to and receives from APPIFY.
R7-MEC-I: MEC should not learn that identifiers "Alice" or IMSI are relevant to the messages.
R8-MEC-I: MEC should not learn that identifier Alice13 is relevant to the messages.
R9-MEC-U: MEC should not be able to distinguish whether two messages are related to the same person or the same IMSI.
R10-MEC-U: MEC should not be able to distinguish whether two messages are related to the same user identifier for a MEC application.
R11-APP-D: APPIFY should not learn anything related to Alice in addition to what can be deduced from information that Alice provides while communicating with APPIFY.
R12-APP-I: APPIFY should not learn that "Alice" or IMSI is related to Alice13.
R13-APP-U: APPIFY should not distinguish whether two messages are coming from the same device.
R14-APP-U: APPIFY should not distinguish whether two messages are related to the same IMSI.
R15-OUT-D: Outsiders should not learn which MEC application Alice is using.
R16-OUT-D: Outsiders should not learn the content of what Alice sends and receives.
R17-OUT-I: Outsiders should not learn that the identifiers "Alice" or IMSI are relevant to Alice.
R18-OUT-I: Outsiders should not learn that identifier Alice13 is relevant to Alice.
R19-OUT-U: Outsiders should not be able to distinguish whether two messages are related to the same MEC application.
R20-OUT-U: Outsiders should not be able to distinguish whether two messages are related to the same user identifier.

3.3.1. Justification of Requirements

The privacy requirements above are defined assuming that the parties are independent, i.e., they do not share information with other parties unless it is necessary for providing services. One example of where the parties are independent is when they are parts of

different companies. In our system model, these parties cooperate with each other, but they should avoid sharing confidential information related to their business and customers.

When two parties are dependent, they may share information with each other, even when it is not necessary to provide services to the user. In this case, some privacy requirements do not apply. Dependent cases are discussed further in Section 3.3.2.

We cannot typically predict how the information will be used in the future. That is why some of our requirements, particularly those concerning unlinkability, may appear to be stringent.

Requirements about MNO (R1–R5)

Alice is one of the subscribers of MNO. While setting up the subscription, Alice may have shared her personal information with MNO. That information that Alice provided, together with the shared identity IMSI, and shared keys, should be enough for MNO to provide services to Alice in our scenario. MNO does not need to know which MEC application Alice is using, the identifiers she shares with the MEC application, or the content she receives from the MEC application.

If MNO can detect which identifiers Alice has or which MEC application Alice uses, then MNO can also figure out how often she uses the application. This information can reveal the interests of Alice and her daily habits. As a result, MNO may provide, for example, unwanted and targeted advertisements or special offers to her.

Requirements about MEC (R6–R10)

MEC delivers messages of Alice from MNO to the MEC application, and vice versa. MEC only needs to know the destination of the messages, i.e., to which MEC application should the messages be transferred. The authorization of the user is performed by MNO and the MEC application. In our scenario, services of MEC do not depend on whether the user is authorized or not. Therefore, MEC should not learn why Alice uses the MEC application or know details about her, such as her real name and her identifiers for MNO and the MEC application.

Requirements about APPIFY (R11–R14)

Alice wants to use the MEC application. Before she starts using the application, it might require authenticating or requesting some information from her. For these purposes, the MEC application should either ask for information directly from Alice or use a third-party authentication service, which both Alice and the MEC application agree with. In any case, the MEC application should not gather any information from any party other than Alice. However, Alice may give the MEC application some details about her, such as her real name or a fake name, home address, and phone number, to improve the user experience with the MEC application. In addition, APPIFY may learn some information due to the nature of the scenario, even though Alice does not provide it herself. One example is the approximate location of Alice because APPIFY finds out that she should be somewhere close to where the MEC is located.

For the quality of the service, the MEC application might want to know what type of device the user has. However, the MEC application should not know which individual device the user has when connecting to it. Furthermore, the MEC application should not relate two messages sent from the same individual device or use the same IMSI for connection to MNO. For example, assume that Bob is another user of the MEC application APPIFY and has the identifier Bob57, while the identifier of Alice is Alice13. Bob may use APPIFY in his IoT device, and this device is connected to the internet through the mobile device of Alice. Therefore, if APPIFY can be used to relate messages to each other based on the device identity or IMSI, it would conclude that Alice13 and Bob57 use the same device or have the same IMSI. Thus, APPIFY can further assume that, with high probability, either these identifiers belong to the same person or these two users have a close relationship with each other. The MEC application does not need to have this kind of knowledge.

Requirements about Outsiders (R15–R20)

MNO, MEC, and the MEC application all take part in providing services to Alice. They need to know some information regarding Alice. However, outsiders are not included in this process. Therefore, all three types of outsiders should learn nothing about Alice, for example, the content or the volume of the usage and the identifiers she shares with MNO and MEC application. They should not know which MEC application Alice uses or who uses APPIFY. In addition, outsiders should not be able to distinguish whether two messages in the network are related to the same MEC application or whether they are sent from the same user or different users.

3.3.2. Privacy Requirements for Dependent Parties

When the parties are dependent, i.e., sharing information that is not strictly needed for providing the service, some of the privacy requirements may not be applicable. The parties may be dependent, for example, when they are parts of the same company.

Next, we list the different combinations of dependent parties and analyze how these combinations affect the privacy requirements. Our model has three parties serving the user: MNO, MEC, and APPIFY. The number of dependency combinations between those three parties is five. (In general, when n parties serve the user, the number of dependency combinations equals the number of partitions of an n-element set, which is given by the Bell number B_n [49]).

Table 1 shows, among other things, the applicability of the requirements in the five dependency combinations between MNO, MEC, and APPIFY. The cells with N/A represent the requirements that are not applicable, while the rest are applicable. Observe, first, that requirements R15–R20 are applicable in all combinations since the system should always be protected against outsiders. Second, all of the requirements, R1–R20, apply when MNO, MEC, and APPIFY are independent. This combination is represented in the column 'NONE' of Table 1. The applicability of the requirements in the other dependency combinations is discussed later in this section.

Each of the three parties, MNO, MEC, and APPIFY, could also be dependent on outsiders. As described in Section 3.2, we have three types of outsiders: entities that are not visible in the system model, other subscribers of MNO, and other MEC applications. We do not consider dependencies with the first two types of outsiders any further. The reason is that they are potentially malicious, and collaborating with them would also make MNO, MEC, or APPIFY malicious. We do not include the third type of outsiders in Table 1 either, but we discuss dependencies with MEC applications other than APPIFY later in this section.

Several users, several applications, or both, could collaborate between themselves in an onion-routing-like manner to improve privacy [28]. For example, Alice wants to use APPIFY in MEC. In order to do this, Alice sends a message to UE1, which belongs to another user; UE1 sends it to UE2, which belongs to a third user; UE2 sends it to APP1 in MEC; APP1 sends the message to APPIFY in MEC. By following this route, the parties in the middle and the outsiders cannot learn or understand that the original message is from Alice and that it is intended for APPIFY in MEC. However, this kind of arrangement would make managing the system more complex and lead to less efficient protocols. Therefore, we assume that there is no systematic collaboration between different users or between different applications.

Another example of collaboration is when an application coordinates its communication towards different users, including fabricated traffic. For example, when Alice wants to initiate a connection with APPIFY in MEC, APPIFY initiates another connection with a similar traffic pattern with another user, e.g., Bob. The APPIFY client in the UE of Bob hides the dummy traffic from the end user. Outsiders and the parties between Alice and APPIFY would observe that both parties use APPIFY and have similar traffic patterns. Therefore, it would be hard for the other parties to understand which one of the users is really using the APPIFY in MEC. It should be noted that it is also possible that both users are intentionally

using the application simultaneously. The downside of this kind of privacy protection measure is the doubling of the network traffic and energy consumption to provide the MEC application service to an individual user. Therefore, we assume that APPIFY is not coordinating actions between its sessions with different users.

Table 1. Effect of dependency combinations on applicability of the privacy requirements and how the solution fulfills them.

	Dependency Combinations				
Requirements	**NONE**	**ALL**	**MNO &MEC**	**MEC &APP**	**MNO &APP**
R1-MNO-D	○	N/A	N/A	○	○
R2-MNO-D	●	N/A	●	●	-
R3-MNO-I	●	N/A	●	●	-
R4-MNO-U	○	N/A	N/A	○	○
R5-MNO-U	○	N/A	○	○	○
R6-MEC-D	●	N/A	●	○	●
R7-MEC-I	●	N/A	N/A	●	●
R8-MEC-I	●	N/A	●	N/A	●
R9-MEC-U	○	N/A	N/A	○	○
R10-MEC-U	○	N/A	○	N/A	○
R11-APP-D	●	N/A	●	●	-
R12-APP-I	●	N/A	●	●	-
R13-APP-U	○	N/A	○	○	-
R14-APP-U	○	N/A	○	○	-
R15-OUT-D	○	○	○	○	○
R16-OUT-D	●	●	●	●	●
R17-OUT-I	●	●	●	●	●
R18-OUT-I	●	●	●	●	●
R19-OUT-U	○	○	○	○	○
R20-OUT-U	●	●	●	●	●

In summary, we assume that each session instance between Alice and APPIFY is run independently of any other session instances between any user and any MEC application.

As pointed out earlier, other MEC applications, which are assumed to be honest but curious, could cooperate with MNO, MEC, or APPIFY. The collaboration could be helpful for gathering extra information about Alice or APPIFY, e.g., when Alice is using another MEC application in parallel to using APPIFY. However, collaboration with other MEC applications would not make any privacy requirements non-applicable.

Combination 1: MNO, MEC, and APPIFY Are Dependent

Assume that MNO deploys its own MEC host to the network. Additionally, MNO has a mobile application for its subscribers. All three parties share information related to users with each other.

As explained above, we assume that neither Alice nor APPIFY collaborates with other users or applications. The messages of Alice pass through MNO and MEC to APPIFY. None of the privacy requirements—except those for outsiders—are applicable when all the parties are dependent because MNO, MEC, and APPIFY can learn and share information

about Alice from these messages, e.g., her identities, the services she uses, the content of the messages.

This combination is shown in Table 1, in the column under the heading 'ALL'.

Combination 2: MNO and MEC Are Dependent

MNO deploys its own MEC host, called MNO-MEC, to the network. In this combination, APPIFY is independent, and it shares information neither with MNO nor with MNO-MEC. The MNO and MNO–MEC are dependent parties, so they share information with each other.

Due to our assumption above, there is no collaboration neither between Alice and the other users nor between APPIFY and the other applications. Therefore, some of the requirements become not applicable because MNO and MNO–MEC share information and can associate it with the information they have. In particular, they can form a one-to-one mapping between messages on the UE–MNO interface and messages on the MEC–APPIFY interface. The requirements R1 and R4 are not applicable because MNO–MEC knows which MEC application Alice uses. Regarding the requirements on MEC, R7 and R9 are not applicable because MNO can provide the identities mentioned in those requirements to MNO–MEC. The rest of the requirements on MNO and MEC are still applicable. Since APPIFY is independent of these parties, all the requirements on APPIFY are applicable.

The effect of this combination on privacy requirements is shown in Table 1, in the column 'MNO and MEC'.

Combination 3: MEC and APPIFY Are Dependent

A company deploys MEC to the mobile network and develops an application MEC–APPIFY. While MEC and MEC–APPIFY share information related to users with each other, MNO does not share information with them.

The requirements associated with MNO are not affected since MNO does not learn any additional information from the other parties. However, MEC might learn some information relevant to the user from MEC–APPIFY. The requirements R8 and R10 are about the identity of the user for the MEC–APPIFY, which MEC–APPIFY can share with MEC. Therefore, these requirements are not applicable. The rest of the requirements for MEC and APPIFY are applicable.

This combination is summarized in Table 1, in the column 'MEC and APPIFY'.

Combination 4: MNO and APPIFY Are Dependent

This combination occurs when MNO develops an application, which we shall call MNO–APPIFY, for MNO subscribers; an independent MEC host is deployed to the mobile network; MNO–APPIFY is one of the applications in the MEC host.

All the privacy requirements are applicable for this combination. Even though MNO and APPIFY both have sensitive information about Alice, there is an independent party between them, which is the MEC host. This independent party could prevent automatic correlation between the messages that MNO sends to MEC and MEC sends to MNO–APPIFY. Therefore, MNO cannot be sure if Alice is using APPIFY, and APPIFY cannot be sure if Alice is the user.

The applicability of the requirements is presented in Table 1 in the column 'MNO and APPIFY'.

We assume that Alice is independent of other parties. A possible exception for the independence of Alice can occur when she is an insider in MNO, MEC, or even APPIFY. Now, some of the requirements are not applicable, while others should still be applicable. However, we do not discuss this exceptional case any further.

Please note that privacy requirements could still be valid even when some of the parties providing the service to Alice are in the same company if that company has an appropriate privacy-preserving policy. Such a policy would be putting barriers between MNO, MEC, and APPIFY and preventing them from sharing information with each other

even though they are part of the same company. For instance, the policy could state that "*subjects are only granted access to the data that is not in conflict with other data they possess*" [50].

4. Results and Discussion

In this section, we propose a solution for the problem stated in Section 3 that aims to provide privacy-preserving communication between Alice and APPIFY. Later, we show how this solution meets the privacy requirements listed in Section 3.3.

4.1. Solution

We assume that the main server of APPIFY has a pair of public verification and private signing keys (VK_{main}, SK_{main}). The main server also has a (potentially self-signed root) certificate for public verification key $cert(VK_{main})$. The main server of APPIFY provides this certificate to the APPIFY in MEC.

The main server of APPIFY issues a certificate $cert(APPIFY)$ for APPIFY in the MEC host. This certificate is given to APPIFY when deployed in the MEC host and will be used to authenticate APPIFY to Alice.

We also assume that MNO has public and private verification and signature key pair (VK_{MNO}, SK_{MNO}). The MNO has a (potentially self-signed root) certificate for public verification key $cert(VK_{MNO})$. The MNO sends $cert(VK_{MNO})$ to its subscribers, including Alice.

The MNO issues a certificate $cert(MEC)$ for MEC when MEC is deployed to the network of MNO. This certificate is signed by SK_{MNO}, and it will be used later for authenticating MEC to Alice.

4.1.1. Registration

Before Alice can access APPIFY in MEC, she has to register to the main server of APPIFY. During the sign up, Alice generates a device-specific asymmetric key pair and chooses her user identity for APPIFY (Alice13) and password (psw). The main server of APPIFY checks whether the user identity is reserved. If yes, then Alice is prompted to pick another user identity. If no, the main server stores Alice13 and psw to its database. The main server of APPIFY issues a certificate $cert(Alice13)$, which includes the information of Alice13 being a valid user and which services Alice is authorized to receive. The main server also sends $cert(VK_{main})$ to Alice. For example, the enrollment over secure transport (EST) protocol can be used for issuing the certificate [51]. Alice can start using APPIFY in MEC, after receiving the certificates $cert(Alice13)$ and $cert(VK_{main})$.

The password is widely used to authenticate the user identity of the application. In our scenario, the password is related to Alice13, not to Alice or the UE of Alice. In addition, the public and private key pairs are valid for just one UE. If Alice wants to use APPIFY with another UE, those key pairs cannot be used. Therefore, the password is a necessary part of the solution.

4.1.2. Subsequent Access to the Main Server of APPIFY

After Alice signs up for the application to the main server of APPIFY, explained above in Section 4.1.1, there could be three reasons for subsequent access to the main server. The first is to renew the certificate $cert(Alice13)$. Alice may require obtaining a new certificate due to the change of the user device, adding a new authorized device, or loss of the previous certificates. For renewing the certificate, Alice needs to sign in by using her user identity Alice13 and password psw.

The second reason is to update service settings and user profile. Alice can use either her existing certificate to sign in or her user identity and password pair for this access. After a successful update, the main server of APPIFY should issue a new certificate $cert(Alice13)$, according to the updated service settings.

The third reason for subsequent access to the main server is that no suitable MEC host is deployed near Alice. If she wants to use APPIFY, but there is no MEC with APPIFY around her, she should receive the service from the main server of APPIFY. Alice can use

either her existing certificate to sign in or her user identity and password pair for this access. In this case, the main server does not need to renew the certificate.

When Alice uses her certificate for signing in to the main server, the access could be done using TLS (transport layer security) with mutual authentication based on the certificates. The certificate $cert(Alice13)$ should be sent to the main server after the TLS handshake to protect the privacy of Alice in this case. Note that Alice receives a certificate $cert(VK_{main})$ for the main server during the registration.

Alice could use her username and password to sign in to the main server. The procedures are similar except that Alice sends her username and password instead of authenticating with a certificate after the server-authenticated TLS session is established.

4.1.3. Access to APPIFY in MEC

Alice can access APPIFY in MEC after completing the registration. If there are no accessible MEC hosts around Alice, she then falls back to receive the services from the main server of APPIFY. In that case, Alice would not be able to experience the benefits of the MEC host. In what follows, we assume that a suitable MEC host is deployed near the user for simplicity.

Similar to the legacy systems, the 5G system includes a secure connection between UE and MNO. In addition to this, we also assume secure channels between MNO and MEC and between MEC and APPIFY, as mentioned earlier in Section 3.1.

Schematic illustration of our solution idea is presented in Figure 3. The solution idea includes two other secure channels and the three channels mentioned earlier. These new channels are authenticated, confidential, and integrity protected. The two secure channels could be called an outer channel, between UE and MEC, and an inner channel, between UE and APPIFY. A part of the inner channel, between UE and MEC, runs inside the outer channel.

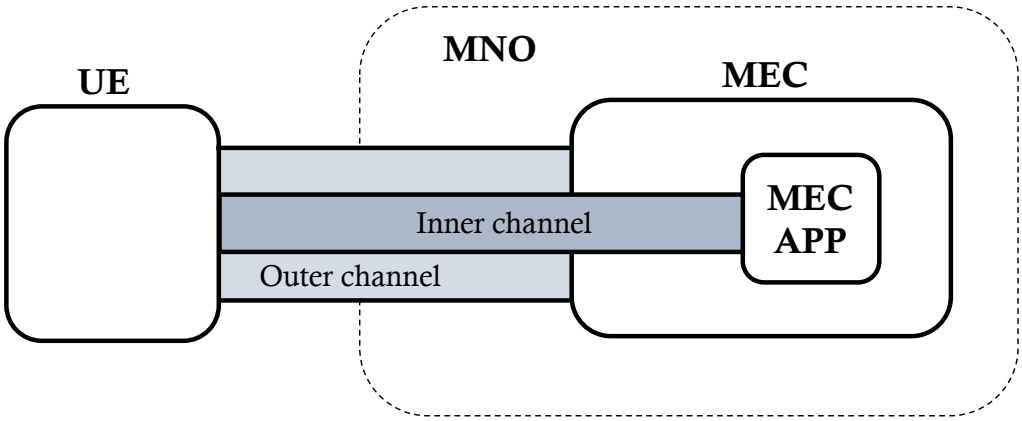

Figure 3. Solution idea for accessing APPIFY in MEC.

Several different protocols can be chosen for the outer and inner channels for realizing our solution. Next, we describe a solution variant which uses datagram transport layer security (DTLS) [52] over user datagram protocol (UDP) for the outer channel and transport layer security (TLS) 1.3 [53] over the transmission control protocol (TCP) for the inner channel.

The DTLS handshake includes server authentication based on the certificate of the server, where the MEC host is the server for the outer channel. The TLS handshake also includes server authentication based on the server certificate, where APPIFY is the server in the case of the inner channel. APPIFY authenticates the client, Alice, after the TLS handshake is completed by using the post-handshake client authentication extension of TLS 1.3 [53]. Therefore, while only Alice authenticates the MEC host in the DTLS connection, in the TLS connection, Alice and APPIFY have mutual authentication.

The solution for accessing APPIFY in MEC is explained step by step and is summarized in Figure 4.

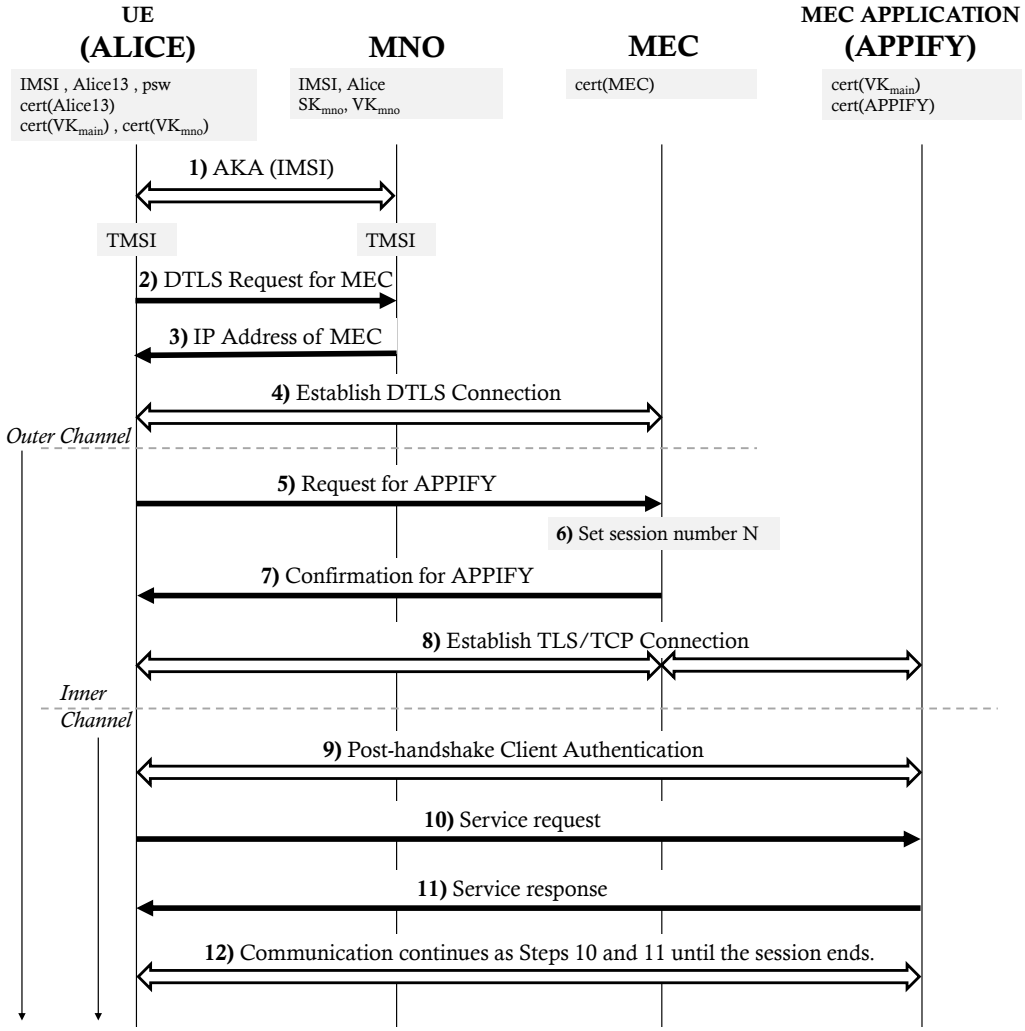

Figure 4. Communication flow of the MEC access.

1. UE of Alice and MNO run the 5G authentication and key agreement (AKA) procedure. As the result of this procedure, a secure connection is established between the UE of Alice and MNO. During the AKA procedure, IMSI is used as an identifier of Alice, and in the end, a temporary identifier TMSI is shared. After this step, the UE of Alice uses TMSI for further identification to MNO.
2. Alice sends a request to MNO for establishing DTLS over UDP connection toward MEC.
3. If there is no deployed MEC in the network of MNO, then MNO replies to Alice with an error message. In this case, Alice connects to the main server of APPIFY. Otherwise, MNO sends the IP address of MEC to Alice.
4. Alice and MEC complete the DTLS handshake. MEC sends $cert(MEC)$ to Alice during the handshake, and Alice verifies this certificate by using the verification key of MNO, VK_{MNO}.
5. Alice sends a request for communication with APPIFY to MEC through the DTLS channel.
6. If there is no APPIFY in MEC, then MEC replies to Alice with an error message. In this case, Alice connects to the main server of APPIFY, similarly as in Step 3. Otherwise, MEC assigns a session number N to the DTLS connection and links it with APPIFY.
7. MEC confirms to the UE that APPIFY is hosted in MEC.

8. Alice sends a request for TLS over the TCP connection with APPIFY through MEC. When MEC receives the request, it inserts the session number N to the message and forwards it to APPIFY. The APPIFY also includes N in the messages directed to Alice in further messages. This way, MEC will deliver the messages to the correct DTLS channels (without the session number). We do not go further into the communication details inside the MEC host.
   APPIFY sends the certificate $cert(APPIFY)$ to the UE of Alice during the TLS handshake. The UE verifies this certificate by using the verification key of the main server of APPIFY, VK_{main}. The TLS connection is established between Alice and APPIFY after successfully completing the handshake.
9. The TLS handshake ensures that Alice is talking with the correct entity. For the mutual authentication, Alice reveals her identity to APPIFY by initiating a post-handshake client authentication with APPIFY. In this phase, Alice sends her certificate $cert(Alice13)$ to APPIFY, and APPIFY verifies it with the verification key of the main server of APPIFY, VK_{main}.
10. Alice sends a service request to APPIFY through the TLS connection.
11. APPIFY replies with the service response to Alice through the TLS connection.
12. The communication between Alice and APPIFY continues as in Steps 10 and 11.

The DTLS and TLS connections stay open during the session. When the session ends, both DTLS and TLS connections are terminated. If Alice13 wants to contact APPIFY again after the session is closed, the procedure starts again from Step 1 or Step 2.

4.2. Analysis

Let us first compare our solution with onion routing. Our solution uses one of the principles of onion routing (OR) [28]: the communication channels between UE and MNO and between MNO and MEC can be viewed as the first and second intermediate hops of an OR path, respectively. On the other hand, an OR path typically includes three or more intermediate nodes that are randomly chosen from a group of onion routers. Our solution has two nonrandom intermediate nodes on the path between Alice and APPIFY: MNO and MEC. Better protection of user privacy, e.g., from colluding MNO and MEC, could be achieved by adding a third node to the path. However, this would also increase the delay in the communication, and the solution would be more complex.

Different variants of the solution would result from choosing different protocols for outer and inner channels. In the variant presented here, DTLS/UDP and TLS/TCP are used for the outer and inner channels, respectively. Reardon et al. [54] show that it is feasible to run TCP on top of DTLS in transport between onion routers: each stream of data has its own TCP connection, but the TCP packets are sent over DTLS protocol, which protects the TCP headers. An advantage of choosing these widely used protocols over custom-made protocols is that the solution is easier to deploy. On the other hand, variants based on custom-made protocols could be more efficient than variants based on standard protocols.

The deployed solution should include measures against denial of service (DoS) attacks. In order to overload the servers, an attacker could try to open many DTLS connections towards the MEC host or TLS connections to APPIFY. Several mechanisms for mitigating DoS attacks exist [52,53,55–57] but we do not discuss those further.

Below, we analyze how the solution in Section 4.1 meets the privacy requirements defined in Section 3.3. We assume that MNO, MEC, and APPIFY are honest but curious throughout our analysis. We start in Section 4.2.1 with the case where all the parties are independent. Then we consider the dependent cases in Section 4.2.2. The summary of the relation between the privacy requirements and the solution is presented in Table 1. As explained in Section 3.3.2, this table shows which of the privacy requirements apply in different dependency cases: if it is written N/A in the cell, then the privacy requirement is not applicable; otherwise, the privacy requirement is applicable. Table 1 also shows how the solution in Section 4.1 fulfills the privacy requirements: a black circle means that the

privacy requirement is fully met, an empty circle means that the privacy requirement is partially fulfilled, and a dash means that the privacy requirement is not fulfilled.

We will see that most of the requirements can be fulfilled by using outer and inner channels. Improved results could be obtained by using more sophisticated mechanisms, e.g., afully-fledged onion routing between the parties. However, as discussed earlier, more sophisticated mechanisms would increase the complexity and cost of the solution, which would decrease the possibility of deployment.

4.2.1. Independent Case

Let us first analyze the case where all three parties, MNO, MEC, and APPIFY, are independent of each other.

Data and Identity Confidentiality

As explained in Section 4.1, we use secure channels between UE and MNO, MNO and MEC, and MEC and APPIFY. Hop-by-hop protection of the path between UE and APPIFY protects the traffic against outsiders. In addition, the messages of other MEC applications might be in the same UE-MNO or MNO-MEC secure channels as APPIFY. However, this does not mean that the other MEC application can extract the communication between Alice and APPIFY. Thus, the requirements R16-OUT-D, R17-OUT-I, R18-OUT-I, and R20-OUT-U are fulfilled. This hop-by-hop protection is not enough to protect against the honest-but-curious parties who are nodes in the traffic path.

The request for APPIFY in Step 5 of the procedure, see Figure 4, and subsequent messages between Alice and APPIFY are delivered inside the DTLS channel. This prevents MNO from learning the identities, APPIFY and Alice13, as well as the content of messages exchanged between Alice and APPIFY. The MNO can only observe that Alice is using some MEC application. We can conclude that the requirements R2-MNO-D and R3-MNO-I are fulfilled for the independent case.

Alice does not send any messages to APPIFY, including her personal information, until the TLS handshake between UE and APPIFY is completed. After the handshake, all the communication between Alice and APPIFY is concealed by the TLS channel. Alice introduces herself as Alice13 by sending $cert(Alice13)$ to APPIFY inside a secure channel to perform the post-handshake client authentication. This way, APPIFY authenticates Alice13, and MEC does not learn the identity of Alice13. The real identity of Alice is not used in any messages of the procedure, and it is only known by MNO. The IMSI is only used between Alice and MNO. Therefore, MEC cannot learn any of these identifiers. We can conclude that the requirements R6-MEC-D, R7-MEC-I, R8-MEC-I are fulfilled.

APPIFY is at the end of the communication path. Since all the information between Alice and APPIFY, similarly as between MEC and APPIFY, is transferred inside an integrity-protected channel, APPIFY receives only the information that other parties have intended for it. Therefore, APPIFY cannot learn more information than what Alice provides. Similarly, as we explained earlier for MEC, we can conclude that APPIFY does not learn either of the identifiers "Alice" and IMSI. Therefore, APPIFY cannot relate these identifiers with Alice13. Thus, the requirements R11-APP-D and R12-APP-I are fulfilled.

It should be noted that after Step 11, Alice might choose to share her IMSI, her real name "Alice", or some other information with APPIFY, e.g., to receive better service. The protocol cannot protect against such disclosure of information. However, the same is probably true for any protocol that allows the free-format exchange of information.

Outsiders and MNO cannot see the identifier of APPIFY because it is carried inside secure channels. Still, traffic analysis may help in identifying the MEC application in use; see [58,59]. For that reason, our solution does not fully meet the requirements R1-MNO-D and R15-OUT-D. Another example of gathering information indirectly is when the other MEC application observes a load on the MEC host. The other MEC application could then correlate the observed load to observed traffic patterns related to a particular user, e.g., on the radio interface. This may help in concluding that the user is connected to APPIFY. If the

application cannot be recognized by studying encrypted traffic patterns or other indirect means, then R1-MNO-D and R15-OUT-D are met.

Unlinkability

The requirements about unlinkability are partially met with our solution. In Step 6, MEC creates a one-to-one mapping from the DTLS session between Alice and MEC to the TLS session between Alice and APPIFY. This means that anybody who can recognize that two messages belong to the same DTLS session (or the same TLS session) also learns that these messages are between the same user and the same MEC application. There are many ways for even an outsider to find out that two protected messages belong to the same (D)TLS session (see, for example, [60]). Therefore, we limit the discussion of unlinkability requirements to the case where the two messages under consideration belong to two different DTLS or TLS sessions.

Our solution generates new DTLS and TLS channels from scratch for every new connection between the user and APPIFY. This makes it more difficult for parties other than Alice and APPIFY to link different TLS channels to each other.

As explained earlier, the MNO cannot identify either APPIFY or Alice13. When a new DTLS connection is established, MNO cannot know whether this connection is for the same MEC application as an earlier connection, except by analyzing traffic patterns. Thus, requirement R4-MNO-U is partially met.

MNO could assume that if the same UE connects to the same MEC application, then the identifier of the user toward that application is also the same. However, MNO cannot know this for sure. For example, the UE could act as a hotspot and be shared by several users. We conclude that the requirement R5-MNO-U is partially met.

Similar reasoning as for MNO and R4-MNO-U above can be carried out for the case of outsiders. Therefore, only by traffic analysis can other users and parties not visible in the system model learn that two UEs use the same MEC application. Similarly, only by traffic analysis can these types of outsiders learn that two messages from two different DTLS sessions relate to the same MEC application. The third type of outsiders, i.e., the other MEC applications, can additionally gather information about the internal state of the MEC host. However, it is still limited as to how well this kind of indirect information could help identify the used MEC application. Therefore, the requirement R19-OUT-U is partially met.

When establishing the outer DTLS channel, Alice authenticates MEC, but MEC does not authenticate Alice. This prevents MEC from linking Alice to APPIFY. In the established session, MEC knows which MEC application is used, but the identities of Alice are not revealed to MEC. However, the IP addresses in the messages might reveal that the same device is used in two different DTLS connections. In that case, these two connections are likely from the same user. Still, MEC cannot be sure about that, as explained above for the case of MNO. We can conclude that the requirements R9-MEC-U and R10-MEC-U are partially fulfilled.

We now discuss the unlinkability requirements R13-APP-U and R14-APP-U for APPIFY in the case where two messages are from two different TLS sessions. In that case, APPIFY can conclude that the user is probably using the same device if the user sends the same certificate to APPIFY in both TLS sessions. On the other hand, having different certificates does not imply that different devices are used. We can conclude that requirement R13-APP-U is partially met.

Typically there is a one-to-one mapping between the user device and IMSI. However, there may be several SIM cards in a single device, or the same SIM card could be moved from one device to another. The APPIFY cannot distinguish between any of these cases. We can conclude that R14-APP-U is partially met.

In summary, our solution meets the privacy requirements well, except for the unlinkability part. An adversary may recognize that messages belong to the same DTLS or TLS session and conduct traffic analysis. However, similar arguments against unlinkability can be made in the setting where applications are used without the help of MEC technology.

4.2.2. Dependent Cases

In Section 3.3.2, we identified which privacy requirements are applicable for the dependency combinations. Now, we analyze how the solution in Section 4.1 meets the applicable privacy requirements for dependent cases.

How our solution fulfills the privacy requirements for outsiders of the first two types (parties non-visible in the system model and other MNO subscribers than Alice) is not affected by the dependency between the parties participating in the protocol. For that reason, our solution fulfills requirements R15–R20 in the case of the first two types of outsiders for all dependency cases in the same way as for the case where all parties are independent.

Let us now consider the third type of outsiders, i.e., the other MEC applications inside the same MEC host, where APPIFY is located. It can be possible that some other MEC applications are dependent on MNO, MEC, or APPIFY, e.g., these parties have their own applications in the MEC host. As an example of how such dependency may have an effect on privacy, assume that Alice establishes two DTLS sessions from the same device, which also has the same IP address, to APPIFY and another MEC application. Suppose the MEC host is dependent on this other MEC application. In that case, MEC knows that the same user is establishing the two DTLS sessions, and the other MEC application knows that its user, say, Alice64, also uses APPIFY with some username. However, the combination of the MEC host and the other MEC application still cannot conclude that this username Alice13, even if they somehow see that Alice13 is active on APPIFY. This example shows that collaboration between the MEC host and some MEC application could reveal some hints about communication between Alice and APPIFY. However, obtaining such indirect information still depends on coincidences, and therefore it does not change substantially how our solution fulfills the requirements about MEC.

The collaboration of MNO and other MEC applications would not provide many benefits because the MEC host is between MNO and the other MEC application.

Finally, if APPIFY collaborates with some other MEC application, these two applications can be considered only one service, where the other MEC application is just an extension of APPIFY.

In summary, none of the possible dependencies between other MEC applications and parties that provide a service to Alice have a substantial impact on how our solution meets the requirements.

The dependence of two out of three parties (MNO, MEC, and APPIFY) would not impact the privacy requirements of the independent party. For example, in the case where MEC and APPIFY are dependent, the requirements for MNO, R1–R5, are fulfilled by the solution similarly as when all three parties are independent. This is because the dependency of MEC and APPIFY does not help MNO to obtain any extra information. Likewise, the dependency of MNO and APPIFY does not change how the privacy requirements for MEC, R6–R10, are fulfilled. Furthermore, the dependency of MNO and MEC does not affect how the solution fulfills the privacy requirements for APPIFY, R11–R14.

MNO and MEC Combination

Combining with MEC does not help MNO, and vice versa, to obtain information that APPIFY has because of the secure channel between Alice and APPIFY in the solution. Therefore, the applicable requirements R2, R3, R5, R6, R8, and R10 are fulfilled similarly to how they are fulfilled in the independent case.

MEC and APPIFY Combination

The requirement R6 is partially fulfilled for this combination because APPIFY knows all the content sent to it but does not know which content is sent by Alice. Combining with APPIFY does not help MEC to obtain information that MNO has, and vice versa, due to the secure channel between Alice and MNO. This is why the requirements R7, R9, R11, R12, and R14 are fulfilled similarly to how they are fulfilled in the independent case. On

the other hand, combining with MEC helps APPIFY to obtain the IP address of the sender, which could help to find out that two messages are coming from the same device. However, the IP address of the device could change, and MNO could allocate the same IP address to another device after some time. Therefore, R13 is still partially fulfilled.

MNO and APPIFY Combination

Combining with APPIFY would clearly help MNO to identify that Alice is using APPIFY, but that would not help to determine which other MEC applications Alice or another user might be using. Therefore, the requirements R1, R4, and R5 are partially fulfilled. APPIFY would also help MNO to identify what Alice is sending to and receiving from APPIFY and help APPIFY to identify IMSI and the device from which it receives messages. Hence, the solution does not fulfill R2, R3, R11, R12, R13, or R14 in this combination. However, MEC could delay and reorder messages to hide which messages from the MNO side relate to the messages on the MEC application side and vice versa. Therefore, we can argue that the solution could be enhanced to partially meet R2, R3, R11, R12, R13, and R14 in this case.

In summary, our solution fulfills the applicable privacy requirements for the MNO and MEC combination in the same way as for the case where MNO, MEC, and APPIFY are independent. Additionally, in the case where MEC and APPIFY share information, the requirements are fulfilled similarly to the independent case, except for one requirement, which is still partially fulfilled in the dependent case. However, when MNO and APPIFY share information, our solution does not fulfill six of the privacy requirements unless the solution is enhanced for the MEC host.

Sharing information between honest-but-curious parties increases their power to attack the user privacy while remaining honest but curious. Thus, a protocol that fulfills a set of privacy requirements when the parties are independent does not necessarily fulfill them in the dependent cases.

5. Final Remarks

We have considered privacy challenges when using multi-access edge computing (MEC) applications in 5G in the basic scenario, where a user wants to access a MEC application in a MEC host that is deployed in a network of a mobile network operator (MNO). Since the network entities in this scenario may have different owners, the privacy of the MEC user should be protected not only against external attackers (outsiders) but also against entities that serve the user.

We defined a system model of this situation and then determined an adversary model and privacy requirements for the entities in the system model. We then introduced a privacy-preserving solution for accessing a MEC application and analyzed how this solution meets the privacy requirements.

The defined system model can be applied to cases other than the one that includes MEC, e.g., three parties provide services to Alice. These possibilities are to be considered in future work.

Formal verification and formal modeling of our solution are left for future work. The future work also includes the experiments of our solution through simulation or other means of concrete examples of executing APPIFY by Alice and privacy requirements when the network is under attack, as well as the numerical results of such experiments. Another direction for future work is to protect user privacy in more complicated scenarios, such as (i) where the user switches from one MEC host to another while moving and (ii) roaming situations involving more than one MNO. Privacy-preserving protocols could be developed and analyzed for those scenarios.

Author Contributions: Conceptualization, G.A., P.G. and V.N.; Supervision, P.G. and V.N.; writing—original draft, G.A.; writing—review and editing, P.G. and V.N. All authors have read and agreed to the published version of the manuscript.

Funding: This research is a result of funded collaboration project between University of Helsinki and Huawei Technologies.

Acknowledgments: The authors would like to thank Sami Kekki and Andrey Krendzel for helpful comments.

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

Abbreviations

The following abbreviations are used in this manuscript:

3GPP	3rd Generation Partnership Project
5G	Fifth Generation
AKA	Authentication and Key Agreement
AF	Application Function
DoS	Denial of Service
DTLS	Datagram Transport Layer Security
EST	Enrollment over Secure Transport
ETSI	European Telecommunication Standards Institute
GUTI	Globally Unique Temporary Identifier
IMSI	International Mobile Subscriber Identity
IoT	Internet of Things
MEC	Multi-Access Edge Computing
MNO	Mobile Network Operator
NFV	Network Function Virtualization
OR	Onion Routing
SDN	Software-Defined Networking
SUPI	Subscription Permanent Identifier
TCP	Transmission Control Protocol
TLS	Transport Layer Security
TMSI	Temporary Mobile Subscriber Identity
UDP	User Datagram Protocol
UE	User Equipment
UPF	User Plane Function
V2X	Vehicle-to-Everything
VPN	Virtual Private Network

References

1. Taleb, T.; Samdanis, K.; Mada, B.; Flinck, H.; Dutta, S.; Sabella, D. On Multi-Access Edge Computing: A Survey of the Emerging 5G Network Edge Cloud Architecture and Orchestration. *IEEE Commun. Surv. Tutor.* **2017**, *19*, 1657–1681. [CrossRef]
2. Giust, F.; Verin, G.; Antevski, K.; Chou, J.; Fang, Y.; Featherstone, W.; Fontes, F.; Frydman, D.; Li, A.; Manzalini, A.; et al. MEC Deployments in 4G and Evolution Towards 5G. *White Paper* **2018**, *24*, 1–24.
3. Pham, Q.V.; Fang, F.; Ha, V.N.; Piran, M.J.; Le, M.; Le, L.B.; Hwang, W.J.; Ding, Z. A Survey of Multi-Access Edge Computing in 5G and Beyond: Fundamentals, Technology Integration, and State-of-the-Art. *IEEE Access* **2020**, *8*, 116974–117017. [CrossRef]
4. 3GPP. *5G System Enhancements for Edge Computing*; Technical Specification TS 23.548 V17.1.0; 3GPP: Valbonne, France, 2021.
5. Parada, C.; Fontes, F.; Marques, C.; Cunha, V.; Leitao, C. Multi-Access Edge Computing: A 5G Technology. In Proceedings of the 2018 European Conference on Networks and Communications (EuCNC), Ljubljana, Slovenia, 18–21 June 2018; IEEE: Ljubljana, Slovenia, 2018; pp. 277–279. [CrossRef]
6. Mahbub, M.; Apu Gazi, M.S.; Arabi Provat, S.A.; Islam, M.S. Multi-Access Edge Computing-Aware Internet of Things: MEC-IoT. In Proceedings of the 2020 Emerging Technology in Computing, Communication and Electronics (ETCCE), Dhaka, Bangladesh, 21–22 December 2020; pp. 1–6. [CrossRef]
7. Naouri, A.; Wu, H.; Nouri, N.A.; Dhelim, S.; Ning, H. A Novel Framework for Mobile-Edge Computing by Optimizing Task Offloading. *IEEE Internet Things J.* **2021**, *8*, 13065–13076. [CrossRef]
8. Sun, Q.; Xu, J.; Ma, X.; Zhou, A.; Hsu, C.H.; Wang, S. Edge-Enabled Distributed Deep Learning for 5G Privacy Protection. *IEEE Netw.* **2021**, *35*, 213–219. [CrossRef]
9. Zhu, R.; Liu, L.; Song, H.; Ma, M. Multi-Access Edge Computing Enabled Internet of Things: Advances and Novel Applications. *Neural Comput. Appl.* **2020**, *32*, 15313–15316. [CrossRef]
10. Zanzi, L.; Cirillo, F.; Sciancalepore, V.; Giust, F.; Costa-Perez, X.; Mangiante, S.; Klas, G. Evolving Multi-Access Edge Computing to Support Enhanced IoT Deployments. *IEEE Commun. Stand. Mag.* **2019**, *3*, 26–34. [CrossRef]

11. ETSI. *Multi-Access Edge Computing (MEC); V2X Information Service API*; Group Specification GS MEC 030 V2.1.1; ETSI: Sophia Antipolis, France, 2020.
12. Porambage, P.; Okwuibe, J.; Liyanage, M.; Ylianttila, M.; Taleb, T. Survey on Multi-Access Edge Computing for Internet of Things Realization. *IEEE Commun. Surv. Tutor.* **2018**, *20*, 2961–2991. [CrossRef]
13. Kekki, S.; Featherstone, W.; Fang, Y.; Kuure, P.; Li, A.; Ranjan, A.; Purkayastha, D.; Feng, J.; Frydman, D.; Verin, G.; et al. *MEC in 5G Networks*; White Paper 28; ETSI: Sophia Antipolis, France, 2018.
14. Hammer, J.; Moll, P.; Hellwagner, H. Transparent Access to 5G Edge Computing Services. In Proceedings of the 2019 IEEE International Parallel and Distributed Processing Symposium Workshops (IPDPSW), Rio de Janeiro, Brazil, 20–24 May 2019; pp. 895–898. [CrossRef]
15. Meir, M. What Is a Third-Party Service Provider? 2021. Available online: https://securityscorecard.com/blog/what-is-a-third-party-service-provider (accessed on 17 January 2022).
16. ETSI. *Multi-Access Edge Computing (MEC); Phase 2: Use Cases and Requirements*; Group Specification GS MEC 002 V2.1.1; ETSI: Sophia Antipolis, France, 2018.
17. Dresch, A.; Lacerda, D.P.; Antunes, J.A.V., Jr. *Design Science Research: A Method for Science and Technology Advancement*; Springer International Publishing: Cham, Switzerland, 2015. [CrossRef]
18. Peffers, K.; Tuunanen, T.; Rothenberger, M.A.; Chatterjee, S. A Design Science Research Methodology for Information Systems Research. *J. Manag. Inf. Syst.* **2007**, *24*, 45–77. [CrossRef]
19. Ranaweera, P.; Jurcut, A.D.; Liyanage, M. Survey on Multi-Access Edge Computing Security and Privacy. *IEEE Commun. Surv. Tutor.* **2021**, *23*, 1078–1124. [CrossRef]
20. University of Delaware. Managing Data Confidentiality. 2020. Available online: https://www1.udel.edu/security/data/confidentiality.html (accessed on 15 December 2021).
21. 3GPP. *3G Security; Security Architecture*; Technical Specification TS 33.102 V16.0.0; 3GPP: Valbonne, France, 2020.
22. Paverd, A.; Martin, A.; Brown, I. *Modelling and Automatically Analyzing Privacy Properties for Honest-but-Curious Adversaries*; Technical Report; University of Oxford: Oxford, UK, 2014.
23. Alshalan, A.; Pisharody, S.; Huang, D. A Survey of Mobile VPN Technologies. *IEEE Commun. Surv. Tutor.* **2016**, *18*, 1177–1196. [CrossRef]
24. Singh, K.K.V.V.; Gupta, H. A New Approach for the Security of VPN. In Proceedings of the Second International Conference on Information and Communication Technology for Competitive Strategies—ICTCS '16, Udaipur, India, 4–5 March 2016; pp. 1–5. [CrossRef]
25. Sawalmeh, H.; Malayshi, M.; Ahmad, S.; Awad, A. VPN Remote Access OSPF-based VPN Security Vulnerabilities and Counter Measurements. In Proceedings of the 2021 International Conference on Innovation and Intelligence for Informatics, Computing, and Technologies (3ICT), Zallaq, Bahrain, 29–30 September 2021; pp. 236–241. [CrossRef]
26. Goldschlag, D.M.; Reed, M.G.; Syverson, P.F. Hiding Routing Information. In *Proceedings of the First International Workshop on Information Hiding, Cambridge, UK, 30 May–1 June 199*; Springer: Berlin/Heidelberg, Germany, 1996; pp. 137–150.
27. Buccafurri, F.; De Angelis, V.; Idone, M.F.; Labrini, C.; Lazzaro, S. Achieving Sender Anonymity in Tor against the Global Passive Adversary. *Appl. Sci.* **2022**, *12*, 137. [CrossRef]
28. Chauhan, M.; Singh, A.K.; Komal. Survey of Onion Routing Approaches: Advantages, Limitations and Future Scopes. In *Proceeding of the International Conference on Computer Networks, Big Data and IoT (ICCBI—2019), Madurai, India, 19–20 December 2019*; Pandian, A.P., Palanisamy, R., Ntalianis, K., Eds.; Springer International Publishing: Cham, Switzerland, 2020; Volume 49, pp. 686–697. [CrossRef]
29. Ranaweera, P.; Jurcut, A.; Liyanage, M. MEC-enabled 5G Use Cases: A Survey on Security Vulnerabilities and Countermeasures. *ACM Comput. Surv.* **2021**, *54*, 1–37. [CrossRef]
30. 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]
31. Okwuibe, J.; Liyanage, M.; Ahmad, I.; Ylianttila, M. Cloud and MEC Security. In *A Comprehensive Guide to 5G Security*; Liyanage, M., Ahmad, I., Abro, A.B., Gurtov, A., Ylianttila, M., Eds.; John Wiley & Sons, Ltd.: Chichester, UK, 2018; pp. 373–397. [CrossRef]
32. Kim, Y.; Park, J.G.; Lee, J.H. Security Threats in 5G Edge Computing Environments. In Proceedings of the 2020 International Conference on Information and Communication Technology Convergence (ICTC), Jeju, Korea, 21–23 October 2020; pp. 905–907. [CrossRef]
33. Ranaweera, P.; Jurcut, A.D.; Liyanage, M. Realizing Multi-Access Edge Computing Feasibility: Security Perspective. In Proceedings of the 2019 IEEE Conference on Standards for Communications and Networking (CSCN), Granada, Spain, 28–30 October 2019; pp. 1–7. [CrossRef]
34. He, X.; Jin, R.; Dai, H. Deep PDS-Learning for Privacy-Aware Offloading in MEC-Enabled IoT. *IEEE Internet Things J.* **2019**, *6*, 4547–4555. [CrossRef]
35. Lee, J.; Kim, D.; Park, J.; Park, H. A Multi-Server Authentication Protocol Achieving Privacy Protection and Traceability for 5G Mobile Edge Computing. In Proceedings of the 2021 IEEE International Conference on Consumer Electronics (ICCE), Las Vegas, NV, USA, 10–12 January 2021; pp. 1–4. [CrossRef]

36. Zhang, P.; Durresi, M.; Durresi, A. Mobile Privacy Protection Enhanced with Multi-access Edge Computing. In Proceedings of the 2018 IEEE 32nd International Conference on Advanced Information Networking and Applications (AINA), Krakow, Poland, 16–18 May 2018; pp. 724–731. [CrossRef]
37. Kotulski, Z.; Niewolski, W.; Nowak, T.W.; Sepczuk, M. New Security Architecture of Access Control in 5G MEC. In *Security in Computing and Communications*; Thampi, S.M., Wang, G., Rawat, D.B., Ko, R., Fan, C.I., Eds.; Communications in Computer and Information Science; Springer: Singapore, 2021; Volume 1364, pp. 77–91. [CrossRef]
38. Khan, R.; Kumar, P.; Jayakody, D.N.K.; Liyanage, M. A Survey on Security and Privacy of 5G Technologies: Potential Solutions, Recent Advancements, and Future Directions. *IEEE Commun. Surv. Tutor.* **2020**, *22*, 196–248. [CrossRef]
39. Ahmad, I.; Kumar, T.; Liyanage, M.; Okwuibe, J.; Ylianttila, M.; Gurtov, A. Overview of 5G Security Challenges and Solutions. *IEEE Commun. Stand. Mag.* **2018**, *2*, 36–43. [CrossRef]
40. Carvalho, G.H.S.; Woungang, I.; Anpalagan, A.; Traore, I. When Agile Security Meets 5G. *IEEE Access* **2020**, *8*, 166212–166225. [CrossRef]
41. Ojanpera, T.; Berg, H.V.D.; IJntema, W.; Schwartz, R.D.S.; Djurica, M. Application Synchronization Among Multiple MEC Servers in Connected Vehicle Scenarios. In Proceedings of the 2018 IEEE 88th Vehicular Technology Conference (VTC-Fall), Chicago, IL, USA, 27–30 August 2018; pp. 1–5. [CrossRef]
42. Do, Q.; Martini, B.; Choo, K.K.R. The role of the adversary model in applied security research. *Comput. Secur.* **2019**, *81*, 156–181. [CrossRef]
43. Rice, T.; Seppala, G.; Edgar, T.W.; Cain, D.; Choi, E. Fused Sensor Analysis and Advanced Control of Industrial Field Devices for Security: Cymbiote Multi-Source Sensor Fusion Platform. In Proceedings of the Northwest Cybersecurity Symposium, Richland, WA, USA, 8–10 April 2019; pp. 1–8. [CrossRef]
44. Halpern, J.Y.; Pucella, R. Modeling Adversaries in a Logic for Security Protocol Analysis. In *Formal Aspects of Security*; Goos, G., Hartmanis, J., van Leeuwen, J., Abdallah, A.E., Ryan, P., Schneider, S., Eds.; Springer: Berlin/Heidelberg, Germany, 2003; Volume 2629, pp. 115–132. [CrossRef]
45. Moradi, A.; Venkategowda, N.K.D.; Pouria Talebi, S.; Werner, S. Distributed Kalman Filtering with Privacy against Honest-but-Curious Adversaries. In Proceedings of the 2021 55th Asilomar Conference on Signals, Systems, and Computers, Pacific Grove, CA, USA, 31 October–3 November 2021; pp. 790–794. [CrossRef]
46. Herzog, J. A computational interpretation of Dolev–Yao adversaries. *Theor. Comput. Sci.* **2005**, *340*, 57–81. [CrossRef]
47. Information Commissioner's Office (ICO). Estate Agency Fined £80,000 for Failing to Keep Tenants' Data Safe. 2019. Available online: https://ico.org.uk/about-the-ico/news-and-events/news-and-blogs/2019/07/estate-agency-fined-80-000-for-failing-to-keep-tenants-data-safe/ (accessed on 15 December 2021).
48. Freedman, M. How Businesses Are Collecting Data (And What They're Doing With It). 2020. Available online: https://www.businessnewsdaily.com/10625-businesses-collecting-data.html (accessed on 15 December 2021).
49. Guichard, D. Bell numbers. In *An Introduction to Combinatorics and Graph Theory*; Creative Commons: San Francisco, CA, USA, 2014; pp. 21–27.
50. Van Tilborg, H.C.A.; Jajodia, S. (Eds.) Commercial Security Model. In *Encyclopedia of Cryptography and Security*; Springer: Boston, MA, USA, 2011; pp. 224–224. [CrossRef]
51. Turner, S. *EST (Enrollment over Secure Transport) Extensions*; IETF RFC 8295; RFC Editor: Wilmington, DE, USA, 2018. [CrossRef]
52. Rescorla, E.; Modadugu, N. *Datagram Transport Layer Security Version 1.2*; IETF RFC 6347; RFC Editor: Wilmington, DE, USA, 2012. [CrossRef]
53. Rescorla, E. *The Transport Layer Security (TLS) Protocol Version 1.3*; IETF RFC 8446; RFC Editor: Wilmington, DE, USA, 2018. [CrossRef]
54. Reardon, J.; Goldberg, I. Improving tor using a TCP-over-DTLS tunnel. In Proceedings of the 18th Conference on USENIX Security Symposium, Montreal, QC, Canada, 10–14 August 2009; USENIX Association: Montreal, QC, Canada, 2009; pp. 119–134. [CrossRef]
55. Rescorla, E.; Dierks, T. *The Transport Layer Security (TLS) Protocol Version 1.2*; IETF RFC 5246; RFC Editor: Wilmington, DE, USA, 2008. [CrossRef]
56. Feng, W.C.; Kaiser, E.; Feng, W.C.; Luu, A. The design and implementation of network puzzles. In Proceedings of the IEEE 24th Annual Joint Conference of the IEEE Computer and Communications Societies, Miami, FL, USA, 13–17 March 2005; Volume 4, pp. 2372–2382. [CrossRef]
57. Gusatu, M.; Olimid, R.F. Improved security solutions for DDoS mitigation in 5G Multi-access Edge Computing. *arXiv* **2021**, arXiv:2111.04801.
58. Taylor, V.F.; Spolaor, R.; Conti, M.; Martinovic, I. Robust Smartphone App Identification via Encrypted Network Traffic Analysis. *IEEE Trans. Inf. Forensics Secur.* **2018**, *13*, 63–78. [CrossRef]
59. Saltaformaggio, B.; Choi, H.; Johnson, K.; Kwon, Y.; Zhang, Q.; Zhang, X.; Xu, D.; Qian, J. Eavesdropping on Fine-Grained User Activities within Smartphone Apps over Encrypted Network Traffic. In Proceedings of the 10th USENIX Conference on Offensive Technologies, WOOT'16, Austin, TX, USA, 8–9 August 2016; USENIX Association: Austin, TX, USA, 2016; pp. 69–78. [CrossRef]
60. Pironti, A.; Strub, P.Y.; Bhargavan, K. *Identifying Website Users by TLS Traffic Analysis: New Attacks and Effective Countermeasures, Revision 1*; Research Report; INRIA: Rocquencourt, France, 2012.