Privacy-Preserving Access for Multi-Access Edge Computing (MEC) Applications

Akman, Gizem

Springer, Cham
2021-09-10

Akman , G , Ginzboorg , P & Niemi , V 2021 , Privacy-Preserving Access for Multi-Access Edge Computing (MEC) Applications . in O Gervasi , B Murgante , S Misra , C Garau , I
þÿB l e i   ,   D   T a n i a r   ,   B   O   A p d u h a n   ,   A   M   A   C   R o c h a   ,   E   T a r a n t i n o   &   C   M   T
þÿC o m p u t a t i o n a l   S c i e n c e   a n d   I t s   A p p l i c a t i o n s   I C C S A   2 0 2 1   .   L e c t u r e   N o
Science , vol. 12956 , Springer, Cham , pp. 44-59 , International Conference on
þÿC o m p u t a t i o n a l   S c i e n c e   a n d   I t s   A p p l i c a t i o n s   I C C S A   2 0 2 1   ,   C a g l i a r i   ,   I

http://hdl.handle.net/10138/347912
https://doi.org/10.1007/978-3-030-87010-2_4

acceptedVersion

Privacy-Preserving Access for Multi-Access Edge Computing (MEC) Applications

Gizem Akman[1,2][0000−0001−8752−7232], Philip Ginzboorg[3,4][0000−0003−4579−3668], and Valtteri Niemi[1,2][0000−0002−3228−4904]

[1] University of Helsinki, Helsinki, Finland
[2] Helsinki Institute for Information Technology (HIIT), Helsinki, Finland
{gizem.akman,valtteri.niemi}@helsinki.fi
[3] Huawei Technologies, Helsinki, Finland
[4] Aalto University, Helsinki, Finland
philip.ginzboorg@huawei.com

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 is about 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, then define the adversary model and privacy requirements for this system model. Second, we introduce a privacy-preserving access solution for the system model and analyze the solution against the privacy requirements.

Keywords: MEC · 5G · privacy · unlinkability

1 Introduction and Related Work

The European Telecommunication Standards Institute (ETSI) initiated standardization for bringing cloud computing to the edge of the network in 2014. Since 2017 the name for this initiative is Multi-Access Edge Computing (MEC). Edge computing is described by the 3rd Generation Partnership Project (3GPP) as an enabling technology that hosts operator and third-party services near the access point of the user to provide efficient service with reduced end-to-end latency and load in transportation network [1]. MEC can be deployed either by Mobile Network Operator (MNO) or by private cloud service providers, such as Amazon AWS and Google [2]. Base stations and access points are some examples for possible deployment locations of MEC servers [2–6].

The Fifth Generation (5G) mobile networks are designed with the goal to provide improved quality of service and quality of experience compared to legacy networks. A 5G mobile network is expected to provide high bandwidth (e.g., 10 Gbps) and very low latency (e.g., 1 ms) at a low operational cost. It should also be able to serve a large number of Internet of Things (IoT) devices, which can communicate in real-time effectively and work reliably and affordably [7]. MEC is considered to be a key technology for 5G development because MEC provides an effective solution for the high demands of 5G [8].

MEC is already deployed in 4G [9], but the new architecture and new features of 5G support a new approach for deploying MEC in 5G architecture, specified in [1]. The communication between MEC and the 5G network is provided through the User Plane Function (UPF). A schematic illustration of the network after the deployment of MEC is shown in Fig. 1.

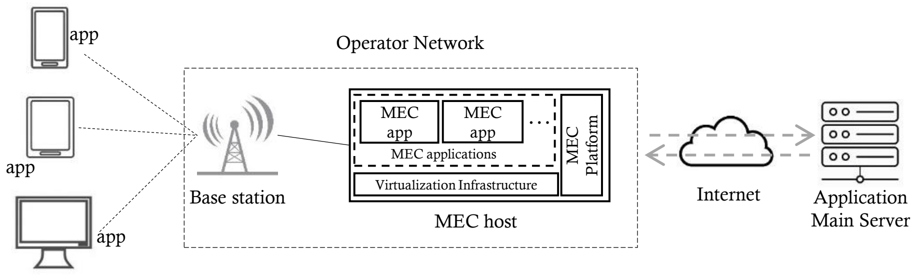

Fig. 1. Overview of network after deployment of MEC, adapted from [3, 10].

Services that may be provided by the MEC applications include computational offloading, distributed content delivery and caching, IoT and big data, smart city services [2], and vehicle-to-everything (V2X) [11].

Privacy concerns arise because MEC usage involves a large amount of data that may be sensitive and personal. The MNO, MEC host, and MEC application may belong to different companies, or they can belong to the same company. Regardless of the ownership, personal data related to the user is passing through MNO and MEC host towards the MEC application. Analysis of the data related to MEC application usage may be useful for various purposes. We discuss this further in Section 3.1.

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

Confidentiality, in general, is "the act of preventing unauthorized entities from reading or accessing sensitive materials" [12]. It is identity confidentiality if the sensitive material is the identity of the user. The user would have several identifiers for MNO and MEC application during the process of using the MEC application. These identifiers should be shared only with the parties who really need to know them. In addition, these identifiers should not be eavesdropped on during the communication on the radio access link [13].

International Mobile Subscriber Identity (IMSI) is the permanent identifier of the subscriber in a mobile network. In legacy mobile networks, the IMSI is sent over the radio access link as rarely as possible, and a randomly generated temporary identifier is sent instead. This randomly generated identifier is called Temporary Mobile Subscriber Identity (TMSI) in 2G and 3G systems, and 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 Subscription Permanent Identifier (SUPI). The 5G network never sends SUPI in the clear over the radio access link. Besides, the 5G system enhances user identity confidentiality with a mechanism, where SUPI can be encrypted by the User Equipment (UE) with the public key of the home network of the subscriber before the UE sends it over the radio access link [1, 14]. For simplicity, we will use the terms IMSI and TMSI to designate the permanent and the temporary identities of the mobile user in the rest of this paper, even when the discussion is about 5G.

Data confidentiality is "protecting data against unintentional, unlawful, or unauthorized access, disclosure, or theft" [15]. Only those parties should have access to the data who really need it.

The definition of unlinkability is given in [16]: "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 focus on the unlinkability of messages and user identities.

Many 5G challenges, including ones related to user privacy, are presented in [17]. The survey [12] explains background information of MEC with security and privacy issues in MEC. The focus in [18] is on MEC for heterogeneous IoT. Privacy issues in MEC are identified, and machine learning privacy is examined for MEC. The survey [19] focuses on the security and privacy of 5G technologies, including MEC. In [20], privacy concerns related to cellular networks are raised, and a solution for protecting the network privacy of the mobile user by using MEC is provided.

Our ideas are related to the concepts of Virtual Private Network (VPN) and Onion Routing (OR). The VPN provides a private and secure channel over the public network. This secure channel is obtained by the tunneling protocol between two entities [22]. Onion Routing provides anonymous communication by routing encrypted traffic over several different routers [23].

Our system model is introduced in Section 2. We define the adversary model in Section 3. Then, we list requirements that will minimize the exposure of private data of the users in Section 4, and introduce a solution for privacy-preserving access to MEC applications in Section 5. We also show how the solution meets privacy requirements. The paper ends with the conclusion and future work in Section 6.

2 System Model

Our system model is based on the following scenario. In the rest of the paper, we denote by "Alice" her real name while Alice without quotes refers to the person. Alice is a subscriber of a Mobile Network Operator, MNO. Alice wants to use a MEC application, which is called APPIFY. In order to use APPIFY, the messages of Alice should go through both the network of MNO and the MEC host. The traffics between Alice and the MNO network, between the MNO network and MEC host, and between the MEC host and APPIFY are secured. However, the MNO and the MEC host can see the message flow between Alice and APPIFY. Still, Alice wants to communicate with APPIFY without revealing the identity of APPIFY and the content of the messages to MNO. Alice also wants to be anonymous towards the MEC host. For example, the MEC host should not learn the IMSI of Alice.

We build an abstract model with the four parties involved in this scenario: Alice, MNO, MEC, and APPIFY. Please note that APPIFY represents a generic MEC application in the model, similarly as Alice takes the role of a generic user. This model is visualized in Fig. 2. Next, the elements of the model are described in more detail.

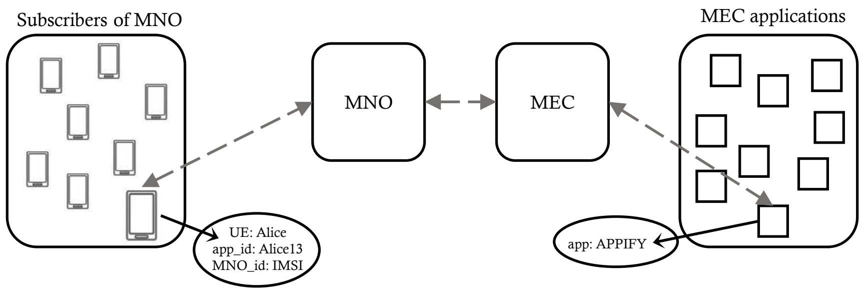

Fig. 2. Elements of the system model.

Mobile Network Operator (MNO) provides services to Alice and other subscribers. IMSI is the identifier of Alice for MNO. In addition to IMSI, MNO may know other information related to Alice, such as her phone number, home address, email address, social security number, etc.

Recall that the MEC host is deployed in MNO, and MEC applications are running in the MEC host. In our model, we call the MEC host simply as MEC. It transfers messages of users between MNO and MEC applications. During this process, MEC does not share any dedicated identifiers with users. Our model does not depend on any specifics of the 5G network, and it also applies to the 4G network.

APPIFY is one of the MEC applications in the MEC host. Typically, the data in the MEC application residing in the MEC host is synchronized regularly with the main server (see Fig. 1) to provide up-to-date service to its users [24]. Both Alice and APPIFY communicate with the main server of APPIFY when it is necessary. For simplicity, we do not include the main server of the application in our model.

There are many other subscribers in addition to Alice; similarly, there are many MEC applications in addition to APPIFY. From the point of view of Alice communicating with APPIFY, these other subscribers and MEC applications are outsiders.

3 Adversary Model

After defining the system model, we now move to security and privacy aspects. First, we identify potential types of adversaries relevant to the model. These types are honest-but-curious adversary and Dolev-Yao adversary.

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" [16]. In our scenario, we assume that MNO, MEC, and APPIFY are honest-but-curious adversaries. They are acting according to the protocol, but on the side, they are passively collecting information.

Dolev-Yao adversary can see, delete, replay, reroute, and reorder all messages in the network [25]. In the model, outsiders are Dolev-Yao adversaries. Other users than Alice may be malicious adversaries, including Dolev-Yao capabilities.

In addition to being honest-but-curious, the parties, MNO, MEC, and APPIFY, are also passive Dolev-Yao adversaries: they may see the messages sent between other parties, but they do not actively interfere with such messages.

3.1 Justification of the Adversary Model

Three parties in the scenario, MNO, MEC, and APPIFY, are assumed to be passive Dolev-Yao and honest-but-curious adversaries. They are passive Dolev-Yao because they are assumed to be able to see all the messages. However, they are honest and do not interfere in the communication by deleting, replaying, rerouting, and reordering the messages. It makes sense to assume that these parties are honest because they do business and provide services for their customers. If they misuse their powers, e.g., by selling customer data to third parties, it would be possible to detect this abnormal behavior in the network. As a result of the malicious action, they can get a bad reputation and lose the trust of the customers; an example case can be found in [26]. Therefore, they are likely to follow the rules.

MNO, MEC, and APPIFY can get curious about their customers while providing services to them. MNO and MEC carry messages between Alice and APPIFY. If MNO and MEC get curious, they can capture the messages they forward. They can learn how the customers use services, which services they prefer, how often they use these applications, and many other details related to customers. So they can analyze the needs of their clients. This analysis provides companies with an understanding of what customers want based on their previous behaviors [27]. Similarly, APPIFY may be interested to learn more about its customers than what they intentionally share with APPIFY.

Outsiders are full Dolev-Yao adversaries. An outsider is any party other than Alice, MNO, MEC, and APPIFY. Outsiders include other users that are subscribers of the same MNO or customers of APPIFY, or both. These other users could be malicious because anybody could get a subscription, and there typically is no reputation that could be lost either.

Alice is not an adversary in the model because it is her identities and data we try to protect.

4 Privacy Requirements

As explained in Section 2, Alice, MNO, MEC, and APPIFY are the parties of the system model. Alice is the user, and the rest are providing services to Alice. Each party sends and receives messages, and these messages include information that is necessary for the service.

Three properties were taken into consideration while determining the privacy requirements: data confidentiality, identity confidentiality, and unlinkability.

Data confidentiality-related requirements, which we label by D, aim to keep information private for parties who need the information. In other words, each party should only know the information that is necessary for them to provide the service and not reveal this information to the other parties. Identity confidentiality-related requirements (I) are for protecting the identities of the user. One party should not learn the identifier by which the other party knows the user because learning this information is not needed by the first party to provide the service. Unlinkability-related requirements (U) aim to prevent distinguishing whether any two messages belong to the same user, same destination, or have the same identifier.

Next, we give the full list of privacy requirements.

R1-MNO-D: MNO should not learn what content Alice sends to and receives from APPIFY.

R2-MNO-D: MNO should not learn which MEC application Alice is using.

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 go 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 identities "Alice", Alice13, or IMSI are relevant to the messages.

R8-MEC-U: MEC should not be able to distinguish whether two messages are related to the same user, same IMSI, or same user identifier for a MEC application.

R9-APP-D: APPIFY should not learn anything related to Alice if Alice does not provide such information.

R10-APP-I: APPIFY should not learn that “Alice” or IMSI is related to Alice13.

R11-APP-U: APPIFY should not distinguish whether two messages are coming from the same device.

R12-APP-U: APPIFY should not distinguish whether two messages are related to the same IMSI.

R13-OUT-D: Outsiders should not learn which MEC application Alice is using.

R14-OUT-D: Outsiders should not learn the content of what Alice is sending and receiving.

R15-OUT-I: Outsiders should not learn anything related to the identities of Alice.

R16-OUT-U: Outsiders should not be able to distinguish whether two UEs use the same MEC application.

R17-OUT-U: Outsiders should not be able to distinguish whether two messages are related to the same MEC application.

In the rest of this section, we justify these requirements. While defining the requirements, it is assumed that the parties are independent, i.e., they do not share information other than what is necessary for interoperability. The parties are typically independent if they belong to different companies. In our system model, these companies work together and use services of each other, but they should not share confidential information related to their business and customers.

If some parties are dependent, i.e., they share more information with each other than what is necessary for interoperability, some of the privacy requirements do not apply. For the most part, this makes it is easier to fulfill the remaining privacy requirements. However, there are also some cases where the solution presented later in this paper does not as such fulfill all of the remaining requirements. Due to the limitation of space, we do not discuss dependent cases further.

Some of our requirements, especially for unlinkability, may look strict. The reason for strictness is that we cannot predict all the ways of how information is used in the future.

Requirements about MNO (R1-R5): Alice is a subscriber of MNO; therefore, her personal information can be found in the database of MNO. The MEC application that Alice is using and the content that she gets are private to Alice. Also, MNO should not learn about Alice13, the identifier of Alice for APPIFY. The MNO should not be able to distinguish whether two messages are sent to the same MEC application or include the same user identifier for the MEC application.

In order to provide services to Alice, MNO does not need to know the application details, such as the name of the MEC application, the messages between Alice and APPIFY, or the identifier for APPIFY. Distinguishing two messages with the same destination or identifier reveals the frequency of the usage of the application. If MNO learns these kinds of information, then, for example, Alice may receive unwanted advertisements or special offers. Therefore, the name of the application, the content, or the frequency of usage should not be learned by MNO.

Requirements about MEC (R6-R8): MEC is transferring messages between MNO and APPIFY, and MEC does not have other tasks in the model. Therefore, there is no need for MEC to know the content of the messages or details of the sender, such as the identifiers of Alice for other parties. It is enough for MEC to know that a subscriber of MNO is using APPIFY.

MNO and APPIFY do the authorization of Alice, and MEC does not need to worry about Alice being unauthorized. MNO checks whether Alice is a valid subscriber, and APPIFY checks whether Alice13 is registered. "Alice" might be known, if necessary, by MNO and APPIFY. However, MEC does not need to know this information in order to operate.

Requirements about APPIFY (R9-R12): The APPIFY in MEC is responsible for providing service to Alice. In order to do this, APPIFY might request information or verification of identity from Alice, which APPIFY either asks directly from Alice or uses a verification method that Alice agrees with. Either way, APPIFY should not request information from MNO or MEC. Alice can provide her real name or a fake name, phone number, location, etc., to APPIFY. However, APPIFY should not learn anything unless Alice provides it herself. There is some information that APPIFY learns automatically because of the setting. An example is the approximate location of Alice since it is close to the location of MEC.

APPIFY should not distinguish whether the messages are coming from the same device or the same IMSI. Let us take a concrete example to highlight this point. Bob is another user with APPIFY identity Bob57. APPIFY may be used in an IoT device of Bob, which is connected to the mobile device of Alice. Now, the IMSI for Bob would be the same IMSI that Alice is using. If APPIFY learns that Alice13 and Bob57 share the same IMSI, then it shows that either the two users are together or the identifiers belong to the same person. APPIFY does not need to know this.

APPIFY might want to learn which type of device Alice and Bob are using if this helps APPIFY to provide its service. However, knowing exactly which devices Alice and Bob are using is not necessary for that purpose. Furthermore, this information can reveal whether one person uses the identifiers Alice13 and Bob57 or two different users are sharing the same device.

Requirements about Outsiders (R13-R17): Unlike MNO, MEC, and APPIFY, outsiders are not involved in providing services to Alice. Therefore, they do not need to learn anything, e.g., the MEC application that Alice is using, the content of this communication, and the identifiers that Alice is using for MNO and APPIFY. In general, outsiders should not be able to distinguish whether two messages of different users are sent to the same MEC application.

5 Privacy-Preserving Access to MEC Application

In this section, we present a solution that aims to provide privacy-preserving communication between Alice and APPIFY. Later, we show how this solution meets privacy requirements.

5.1 Solution

We assume that the main server of APPIFY has public and private verification and signature key pair (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 sends 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 MEC and will be used to authenticate APPIFY to Alice.

We also assume that MNO has public and private verification and signature key pair (VK_{main},SK_{main}). 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 MNO. This certificate will be used later for authenticating MEC to Alice.

Before Alice can access APPIFY in MEC, she has to register with the main server of APPIFY. During the registration, Alice chooses her user id (Alice13) and password (*psw*). The main server of APPIFY issues a certificate $cert(Alice13)$. This certificate includes the information that Alice13 is a valid user and which services Alice is authorized to get. The main server of APPIFY sends the certificates $cert(Alice13)$ and $cert(VK_{main})$ to Alice after the registration is completed successfully.

Once Alice is done with the registration, Alice can start using APPIFY in MEC. When there is no suitable MEC deployed near Alice, the service is provided by the main server of APPIFY, which means that the user cannot take advantage of the proximity of the MEC host. For simplicity, we assume below that a suitable MEC host is deployed near the user.

The 5G system, similarly to the earlier mobile systems, includes a secure channel between UE and MNO. We also assume there are secure channels between MNO and MEC, as well as between MEC and APPIFY.

Our solution includes two additional secure (authenticated, confidential, and integrity-protected) channels: outer and inner. The outer channel is between UE and MEC, and the inner channel is between UE and APPIFY. The UE-to-MEC part of the inner channel runs inside the outer channel. The solution can be realized in several ways, depending on the choice of protocols for those two channels. Below we describe a variant where Datagram Transport Layer Security (DTLS) [28] (run on top of UDP) is used for the outer channel and Transport Layer Security (TLS) 1.3 [29] (run on top of TCP) for the inner channel.

The DTLS handshake includes server authentication based on the certificate of the server, MEC. The TLS handshake includes server authentication based on the certificate of the server, APPIFY. The client, Alice, is authenticated towards TLS server, APPIFY, after the handshake using the Post-Handshake Client Authentication extension of TLS 1.3. As a result, Alice and APPIFY will have a mutually authenticated TLS connection.

The procedure for accessing APPIFY in local MEC is summarized in Fig. 3 and explained step-by-step in the following.

1. The UE of Alice and MNO run 5G Authentication and Key Agreement (AKA) procedure, resulting from which they establish a secure connection. At this point, Alice and MNO share a Temporary Mobile Subscriber Identity (TMSI) for further identification of Alice to MNO.
2. Alice sends a DTLS communication request (over UDP) for MEC to MNO.
3. If MEC is not deployed in MNO, then MNO returns an error message to Alice. In this case, Alice connects to the main server of APPIFY. If MEC is deployed in MNO, then MNO replies with the IP address of MEC.
4. DTLS handshake is done between Alice and MEC. During the handshake, MEC sends $cert(MEC)$ to Alice, and she verifies the certificate with the verification key of MNO, VK_{MNO}. After the handshake is completed successfully, the DTLS connection is established between Alice and MEC.
5. Alice sends a communication request for APPIFY to MEC through the DTLS channel.
6. If MEC does not host APPIFY, then MEC returns an error message to Alice. In this case, Alice begins to establish a connection to the APPIFY main server over the Internet. If MEC does host APPIFY, then MEC sets a session number N and links this number with the DTLS channel and APPIFY.
7. MEC confirms that it hosts APPIFY.
8. After receiving confirmation that MEC hosts APPIFY, Alice opens TLS/TCP connection to APPIFY. Upon receiving this message, MEC inserts its internal label N and forwards the resulting message to APPIFY. The APPIFY will include N in its messages towards Alice. MEC will use N for directing them to the appropriate DTLS channel but strip N before the message goes into the DTLS channel. We do not go further into the details of communication inside the MEC host.
   During the TLS handshake, the APPIFY sends $cert(APPIFY)$ to Alice, and she verifies the certificate with the verification key of the main server of APPIFY, VK_{main}. After the handshake is completed successfully, the TLS connection is established between Alice and APPIFY.
9. At this point, Alice knows that she is talking to the correct entity. Alice reveals her identity to APPIFY by initiating a Post-Handshake Client Authentication with APPIFY. Alice sends $cert(Alice13)$ 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 should be terminated. If Alice13 wants to contact APPIFY again after the session is closed, the procedure should start again from Step 1 or Step 2.

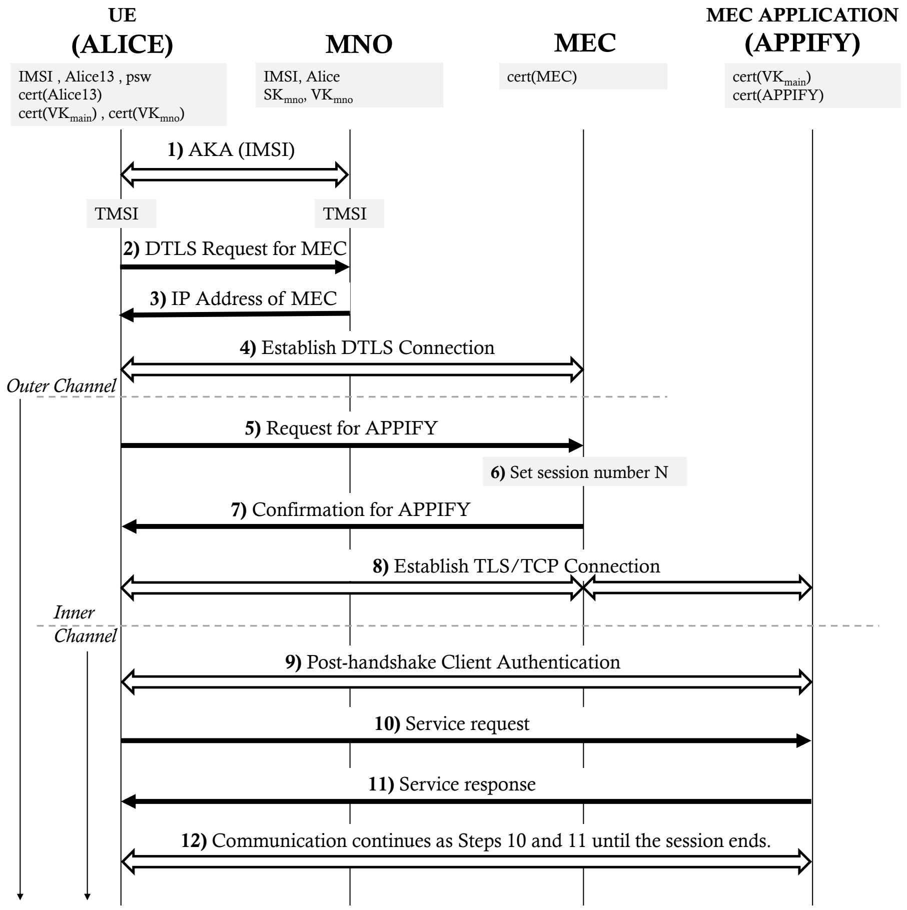

Fig. 3. Communication flow of the MEC access.

5.2 Analysis

Our solution follows one of the principles of Onion Routing (OR) [23]: the communication channels from UE to MNO, and from UE to MEC can be seen as the first and the second routes of OR, respectively. However, a path in OR is random and includes typically (at least) three intermediate nodes, which are chosen from a pool of onion routers. We have a single path with two nodes: MNO and MEC. While adding a third intermediate node on the path between UE and MEC would better protect user privacy against, e.g., colluding MNO and MEC, it would also increase the communication delay and make the solution more complex.

Different protocol choices for outer and inner channels would result in different variants of the solution. The variant presented above, where both outer and inner channels are based on standard, widely used protocols (respectively, DTLS/UDP and TLS/TCP), has the advantage that the solution is easier to deploy than variants based on custom-made protocols. It has been shown that an arrangement where TCP runs on top of DTLS is feasible [31]. On the other hand, a solution based on custom-made protocols could potentially be more efficient than variants based on standard protocols.

The deployed solution needs to have measures against Denial of Service (DoS) attacks that try to overload the servers by opening many DTLS connections towards the MEC host or TLS connections to APPIFY. Various mechanisms to prevent DoS attacks exist [28, 29, 32, 34] but we do not discuss those further.

Now, we analyze how the solution in Section 5.1 meets the privacy requirements defined in Section 4. We will see that we can fulfill most of the requirements by using the outer and inner channels and partially fulfill the rest of the requirements. Better coverage of the requirements could be potentially achieved with more sophisticated mechanisms, e.g., a fully-fledged Onion Routing between the parties. However, this would increase the complexity and cost of the solution, making it less likely to be deployed. Throughout this section, we use the assumption that all three network-side parties are honest-but-curious.

Data and Identity confidentiality: As explained in Section 5.1, 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. Thus, the requirements R14-OUT-D and R15-OUT-I 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 Fig. 3, 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 R1-MNO-D and R3-MNO-I are fulfilled.

Alice does not send any messages to APPIFY, which includes her personal information, until the TLS handshake between UE and APPIFY is completed. After the handshake, all 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 Post-Handshake Client Authentication. This way, APPIFY authenticates Alice13, and MEC does not learn the identity 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 and R7-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 neither APPIFY learns the identifiers "Alice" or IMSI. Therefore, APPIFY cannot relate these identifiers with Alice13. Thus, the requirements R9-APP-D and R10-APP-I are fulfilled.

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 [35, 36]. For that reason, our solution does not fully meet the requirements R2-MNO-D and R13-OUT-D. However, if the application cannot be recognized by studying encrypted traffic patterns, then R2-MNO-D and R13-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 is able to 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 how even an outsider could find out that two protected messages belong to the same (D)TLS session (see, for example, [37]). 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 a new TLS channel from scratch for every new connection between the user and APPIFY. Therefore, only Alice and APPIFY can link different TLS channels to each other.

As explained earlier, the MNO learns neither APPIFY nor 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 towards that application is also the same. However, MNO cannot know this for sure. For example, the UE could be 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, the outsider could learn that two UEs use the same MEC application from traffic analysis only, hence requirement R16-OUT-U is partially met. Also, it is only by traffic analysis that the outsider can learn that two messages from two different DTLS sessions relate to the same MEC application. Therefore, also requirement R17-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 identity of Alice is 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 requirement R8-MEC-U is partially fulfilled.

We now discuss the unlinkability requirements R11-APP-U and R12-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 R11-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, and 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 R12-APP-U is partially met.

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

6 Conclusion and Future Work

This paper sheds light on the privacy challenges related to MEC usage in 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 MNO. We first create a system model of the scenario. Second, we identify an appropriate adversary model covering all parties. Third, privacy requirements are defined for all parties in the system model. Finally, based on those requirements and the adversary model, a privacy-preserving solution is introduced to access MEC application. The solution is analyzed against the privacy requirements.

Future work could include other variants of our solution, formal verification, and user privacy protection in more complicated scenarios, e.g., when the communication parties include multiple MNOs. In particular, in the roaming situation, several MNOs take part in communication.

In addition, there could be complex situations like the following. The subscriber is driving. At some point, the application that the subscriber uses might not anymore be in the MEC host that is deployed by the MNO of the subscriber, but instead, it might be found in the MEC host of another MNO. In these situations, it would be nice if the subscriber can use the MEC hosts of other operators. Privacy-preserving protocols could be developed for those more complicated scenarios.

Acknowledgements

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

References

1. 3GPP TS 23.501 V16.5.0.: System architecture for the 5G System (5GS); Stage 2. Technical Specification, (2020).
2. 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 Communications Surveys & Tutorials **19**(3), pp. 1657–1681 (2017).
3. Pham, Q., Fang, F., Ha, V.N., Piran, M.J., Le, M., Le, L.B., Hwang, W., Ding, Z.: A Survey of Multi-Access Edge Computing in 5G and Beyond: Fundamentals, Technology Integration, and State-of-the-Art. IEEE Access, pp. 1–44 (2020).
4. Mao, Y., You, C., Zhang, J., Huang, K., Letaief, K.B.: A Survey on Mobile Edge Computing: The Communication Perspective. IEEE Communications Surveys & Tutorials **19**(4), pp. 2322–2358 (2017).
5. Parada, C., Fontes, F., Marque, C.S, Cunha, V., Leitão C.: Multi-Access Edge Computing: A 5G Technology. In: European Conference on Networks and Communications (EuCNC), pp. 277–281. IEEE, Ljubljana, Slovenia (2018).
6. Hammer, J., Moll, P., Hellwagner, H.: Transparent Access to 5G Edge Computing Services. In: 2019 IEEE International Parallel and Distributed Processing Symposium Workshops (IPDPSW), pp. 895–898. IEEE, Rio de Janeiro, Brazil (2019).
7. Porambage, P., Okwuibe, J., Liyanage, M., Ylianttila, M., Taleb, T.: Survey on Multi-Access Edge Computing for Internet of Things Realization. IEEE Communications Surveys Tutorials **20**(4), pp. 2961–2991 (2018).
8. Okwuibe, J., Liyanage, M., Ahmad, I., Ylianttila, M.: Cloud and MEC security. A Comprehensive Guide to 5G Security, pp. 373–397. Wiley (2018).
9. ETSI White Paper No.28: MEC in 5G Networks; First Edition, (2018).
10. ETSI GS MEC 002 V2.1.1.: Multi-access Edge Computing (MEC); Phase 2: Use Cases and Requirements. Group Specification, (2018).
11. ETSI GS MEC 030 V2.1.1.: Multi-access Edge Computing (MEC); V2X Information Service API. Group Specification, (2020).
12. Ranaweera, P., Jurcut, A.D., Liyanage, M.: Survey on Multi-Access Edge Computing Security and Privacy. IEEE Communications Surveys & Tutorials, (2021).
13. 3GPP TS 33.102 V16.0.0.: 3G Security; Security architecture. Technical Specification, (2020).
14. 3GPP TS 33.501 V16.3.0.: Security architecture and procedures for 5G system. Technical Specification, (2020).
15. University of Delaware, Secure UD: Managing data confidentiality, `https://www1.udel.edu/security/data/confidentiality.html`. Last accessed 28 July 2020.
16. Paverd, A.J., Martin, A., Brown, I.: Modelling and Automatically Analyzing Privacy Properties for Honest-but-Curious Adversaries. Technical Report (2014).
17. Ahmad, I., Kumar, T., Liyanage, M., Okwuibe, J., Ylianttila, M., Gurtov, A.: Overview of 5G Security Challenges and Solutions. IEEE Communications Standards Magazine **2**(1), pp. 36–43 (2018).

18. 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 Communications Magazine **56**(8), pp. 62–67 (2018).
19. Khan, R., Kumar, P., Jayakody, D., Liyanage, M.: A Survey on Security and Privacy of 5G Technologies: Potential Solutions, Recent Advancements, and Future Directions. IEEE Communications Surveys & Tutorials **22**(1), pp. 196-248 (2020).
20. Zhang, P., Durresi, M., Durresi, A.: Mobile Privacy Protection Enhanced with Multi-access Edge Computing. In: IEEE 32nd International Conference on Advanced Information Networking and Applications (AINA). pp. 724–731. (2018).
21. 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, pp. 1–5. ACM, New York, NY, USA, (2016).
22. Alshalan, A., Pisharody, S., Huang, D.: A Survey of Mobile VPN Technologies. IEEE Communications Surveys & Tutorials **18**(2), pp. 1177–1196 (2016).
23. Chauhan, M., Singh, A.K., Komal: Survey of Onion Routing Approaches: Advantages, Limitations and Future Scopes. In: Proceeding of ICCBI-2019, pp. 686–697. Springer (2020).
24. Ojanperä, T., van den Berg, H., IJntema, W., de Souza Schwartz, R., Djurica, M.: Application Synchronization Among Multiple MEC Servers in Connected Vehicle Scenarios. In: 2018 IEEE 88th Vehicular Technology Conference (VTC-Fall), pp. 1–5. IEEE, Chicago, IL, USA, (2018).
25. Herzog, J.: A computational interpretation of Dolev–Yao adversaries. Theoretical Computer Science **340**(1), pp. 57–81 (2005).
26. ICO (Information Commisioners' Office) News: Estate agency fined £80,000 for failing to keep tenants' data safe, (2009). `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/`. Last Accessed 22 Apr 2021.
27. Freedman, M.: How Businesses Are Collecting Data (And What They're Doing With It), (2020). `https://www.businessnewsdaily.com/10625-businesses-collecting-data.html`. Last Accessed 22 Apr 2021.
28. IETF RFC 6347: Datagram Transport Layer Security Version 1.2, (2012).
29. IETF RFC 8446: The Transport Layer Security (TLS) Protocol Version 1.3, (2018).
30. Paterson, K., van der Merwe, T.: Reactive and Proactive Standardisation of TLS. In: Security Standardisation Research (SSR), pp. 160–186. Royal Holloway, (2016).
31. Reardon, J., Goldberg, I.: Improving Tor using a TCP-over-DTLS Tunnel. In: Proceedings of the 18th conference on USENIX security symposium (SSYM'09). USENIX Association, USA, pp. 119–134, (2009).
32. IETF RFC 5246: The Transport Layer Security (TLS) Protocol Version 1.2, (2008).
33. IETF RFC 7633: X.509v3 TLS Feature Extension, (2015).
34. Feng, W., Kaiser, E., Luu, A.: Design and implementation of network puzzles. In: Proceedings of IEEE Infocom 2005, **4**, pp. 2372-2382 (2005).
35. Taylor, V.F., Spolaor, R., Conti, M., Martinovic, I.: Robust Smartphone App Identification via Encrypted Network Traffic Analysis. IEEE Transactions on Information Forensics and Security, **13**(1), pp. 63–78 (2018).
36. 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), pp. 69–78. USENIX Association, USA, (2016).
37. Pironti. A., Strub, P.Y., Bhargavan, K.: Identifying Website Users by TLS Traffic Analysis: New Attacks and Effective Countermeasures, Rev. 1. INRIA Paris, (2012).