Proxy-Based Federated Authentication: A Transparent Third-Party Solution for Cloud-Edge Federation

Ying-Dar Lin, Duc-Tai Truong, Asad Ali, Chi-Yu Li, Yuan-Cheng Lai, and Thai-Mai Thi Dinh

Abstract

Cloud and Edge computing paradigms provide storage and computing services to traditional and Internet of Things devices. One computing platform is not suitable to fulfill the requirements of all IoT devices because of their heterogeneity. In this regard, a federation of various computing paradigms has been emerging, in which a user (first party) with an account on one computing platform (second party) can access the services provided by another computing platform (third party federated with the first computing platform). The user needs to authenticate itself with the third-party computing platform which does not have user credentials. This work proposes a transparent and standard-compliant proxy-based federated authentication for solving the third-party authentication problem in federated cloud and edge computing paradigms. Transparency allows cloud and edge operators to deploy this proxy without having to change the existing authentication protocols. Experimental results illustrate that, as compared with the concatenation of authentication protocols in cloud and edge, proxy-based federated authentication of edge-to-cloud and cloud-to-edge can reduce the authentication delay time by 27.7 percent and 37.9 percent, respectively, and it is also standard compliant.

Introduction

With the dramatic growth of IoT, the amount of data is increasing explosively and in order to process all that data, computational offloading is an appropriate method [1]. Cloud and Edge are computing platforms, which are suitable solutions for computational offloading. Cloud computing supplies on-demand and broad network access to a shared pool of computing resources. However, a cloud is normally far from the data source and introduces latency in IoT systems. Some of the IoT services may require real-time processing which makes cloud computing inappropriate for those services. Edge computing is more suitable than cloud computing in such cases. Edge computing was developed from the European Telecommunication Standards Institute's (ETSI, https://www.etsi.org/) idea of capabilities of virtualization into the mobile network operators [2]. Edge computing can also provide services as a cloud platform but with less computing power and capacity. Although the edge computing paradigm provides less computing power and capacity than cloud computing, it offers reduced latency as it is closer to IoT devices. Consequently, both cloud and edge play an important role in the deployment of IoT.

A federation of computing platforms then comes into play in order to provide the best of both computing paradigms to the devices. This is a technology which permits service providers to share their virtual infrastructure [3]. Thus, such a federation is like a big computing platform which can satisfy the users' various demands (requirements). A user may access cloud and edge simultaneously for various requirements, such as powerful computation or low latency. The federation allows customers to access the requested service without limitation to each computing paradigm. This federation between cloud and edge then exploits their respective strengths and to reduce their weaknesses. Moreover, a federation allows customers to use the account provided by their home provider to access another provider's service. The federation opens up many new opportunities and challenges, such as user authentication, secure data transfer, and secure software interface, among which authentication is the most important one. In the federation, a user may access a number of applications of providers without having an account on those computing platforms. This, however, leads to one problem: how can providers authenticate user identity without a user account? Third-party authentication [4] is the solution to this problem and relies on the trust between service provider (i.e., foreign service provider) and an authentication provider (i.e., home service provider). If the third-party authentication process is able to confirm the authentication of users successfully, users are also authenticated to the service provider. As a federation can be established between different computing platforms, in this research, we consider a federation proxy between these platforms to provide third-party federated authentication between a foreign service provider and a home service provider.

In this work, we consider all possible federation scenarios and provide a solution for cases when the user account is in either edge or cloud. The main idea is to glue existing authentication protocols in cloud and edge to support both environments. We propose a federation proxy for gluing the protocols. Simulation results show that our

Digital Object Identifier:
10.1109/MNET.011.2000136

*Ying-Dar Lin, Truong Duc Tai, Asad Ali and Chi-Yu Li are with NCTU, Taiwan; Yuan-Cheng Lai is with NTUST, Taiwan;
Thai-Mai Thi Dinh is with VNU University of Engineering and Technology, Vietnam National University, Hanoi, Vietnam.*

0890-8044/20/$25.00 © 2020 IEEE

proposed federation proxy provides transparent and seamless authentication. The rest of the article is organized as follows. First, we review the related work and present the main challenges for a cloud-edge federation. Then we explain the proposed federated proxy solution for the problem, and provide the results. In conclusion, we introduce some suggestions for future work.

Related Work

There are two strategies for authentication in a federation of cloud and edge. One is to create a completely new protocol for third-party federated authentication. However, this is impractical since service providers will not be willing to change their commercial systems. Another option is to combine existing authentication protocols in the cloud with the authentication protocols in the edge that are governed by the 3rd Generation Partnership Project (3GPP, https://www.3gpp.org/) and glue them together so that the changes to the existing cloud and edge system are minimized. The existing protocols for authentication in the cloud and edge (as governed by 3GPP) are OpenID Connect (OIDC) and Evolved Packet System-Authentication and Key Agreement (EPS-AKA). Several studies have researched federated authentication protocols, among which few works [5-9] are about authentication in a cloud-cloud federation, while some are about the federated authentication in cloud and edge. A comparison of protocols used for authentication in cloud and edge federated environments is provided in Table 1. These studies are compared on the basis of their objectives, the protocols involved, considered scenarios, transparency and whether they use a middlebox approach or not. The objectives of some of these studies are security and mobility as well, but they all consider federated authentication. The scenario column shows the computing platforms which are considered in the federation: C-C means cloud to cloud federation, and E-C means edge to cloud federation scenario where the user has an account on edge and C-E means cloud to edge federation scenario where the user has an account on the cloud. The middlebox approach column indicates whether or not the listed studies consider a middlebox approach in their solutions. A middlebox reduces the deployment costs and is easy to install as it does not require any modifications to existing protocols or infrastructure, and it also provides transparency. It can be seen that all the documented studies do not use a middlebox approach and do not provide transparency contrary to our proposed protocol. Wantanabe and Tanaka [10] introduced a federation method relying on OpenID (protocol that allows usage of a single set of user credentials for accessing multiple sites) and a cellular phone to guard user privacy. The user's cellular phone communicates with the User Equipment (UE), which wants to access the service. In this case, the UE is a computer. They designed a federated authentication provider to process the authentication request of the UE sent from the cell phone. The limitation of this work is the need of the manual steps between the UE and the cellular phone. In addition, this work makes changes in the original protocol of OIDC which introduce delay in the authentication process.

Works on SSO	Objective	Protocols	Middle box approach	Transparent	Scenario
[10]	Security	OpenID	No	No	E-C
[11]	Seamless auth.	SAML, GBA	No	No	E-C
[12]	Seamless auth.	New protocol	No	No	E-C
[13]	Security	OAuth2.0	No	No	C-C
[14]	Support mobility	Kerberos	No	No	C-C
[15]	Security	OIDC	No	No	C-C
Ours	Seamless auth.	OIDC, EPSAKA	Yes	Yes	C-E, E-C

TABLE 1. Comparison of third-party authentication protocols between cloud and edge.

Friese *et al*. [11] considered the use cases and technical approach for authentication from IP Multimedia Subsystem (IMS) to Internet identity. In the article, the authors proposed a Single Sign On (SSO) protocol, which is a combination of 3GPP Generic Bootstrapping Architecture (GBA) and Security Assertion Markup Language (SAML). IMS authentication, in this case, is reused to be the third-party authentication on the Internet. However, it still lacks third-party authentication with the 3GPP services, which are not based on IMS architecture. In addition, they did not deal with how a user is verified without Universal Subscriber Identity Module (USIM), even though the non-3GPP access would be popular in a 5G environment. Furthermore, no specific solution was addressed in the article for the mismatch issue when integrating two authentication protocols from different computing platforms. The solution we propose is a transparent proxy-based federated authentication approach for a cloud and edge federation using existing authentication protocols. The proposed solution does not require any changes to the existing protocols and is better than other existing federated authentication mechanisms.

Problems: Cloud-to-Edge and Edge-to-Cloud

In this section, we will explain the problem and authentication model in detail. Our authentication model consists of three parties: the user as the first party, a home service provider as the second, and a foreign service provider as the third. Our purpose is to provide authentication to the first party in order to access the services provided by the third party by using the account of the second party. The user account can either be in edge or cloud which gives us two scenarios for authentication, as shown in Figs. 1a and 1b. In the first scenario (Fig. 1a) edge is the third party, which contains the service that the user desires. A cloud service provider is the second party, which stores the user's credentials. The edge and cloud are assumed to be federated with each other, and the user has no account on the edge, but desires to access a service here. The objective is to provide edge services to the user who does not have any account there but is a subscriber to the cloud. We call this the cloud-to-edge scenario. The second scenario (Fig. 1b) is similar to the first, but now the edge and cloud are the second and third parties, respectively. In this scenario, the user has no account on the cloud, but wants to use services on

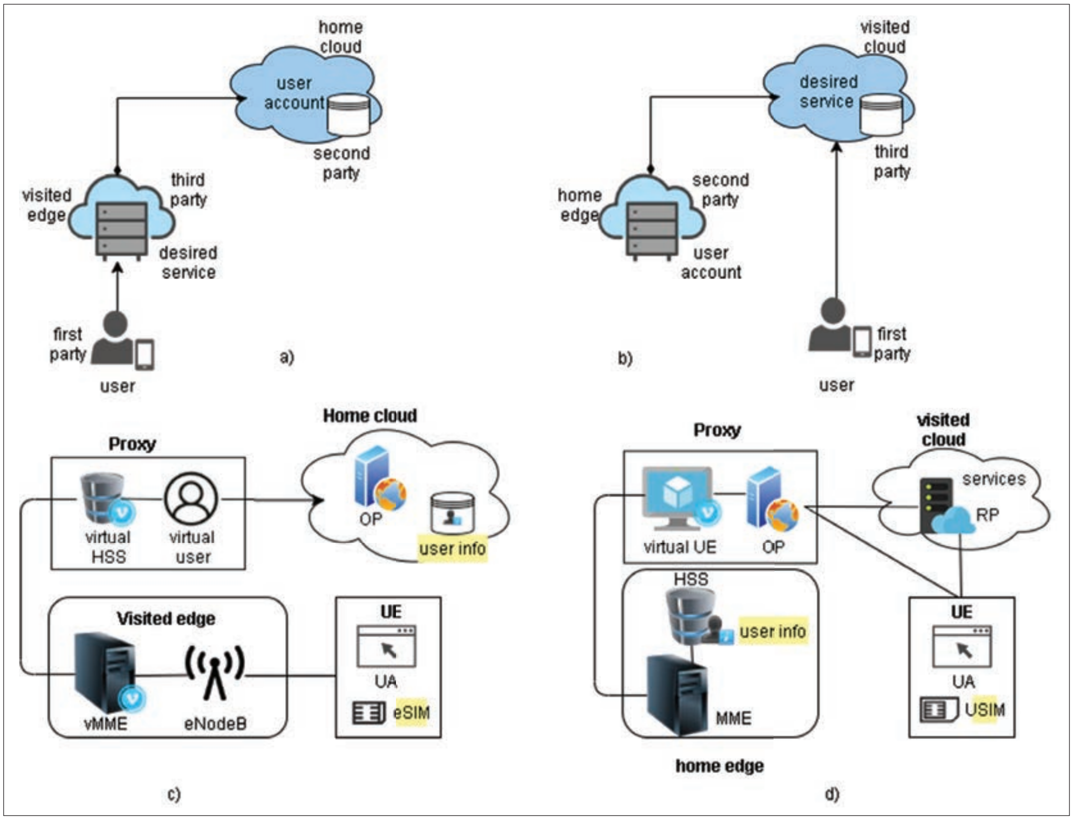

FIGURE 1. a) Third-party authentication problem cloud-to-edge scenario; b) third-party authentication problem edge-to-cloud scenario; c) third-party authentication solution cloud-to-edge scenario; d) third-party authentication solution cloud-to-edge scenario.

this cloud. The user, in this case, has an account only in the edge. Hence, the objective is to provide cloud services to the user who is a subscriber of an edge deployed by some cellular network provider. We call this the edge-to-cloud scenario. The cloud-to-edge scenario might be unpopular, but it is necessary to consider this scenario as such a connection is possible in a federation.

There are multiple issues that arise from these scenarios, such as: The authentication of the user between edge and cloud service provider, design of communication intermediary between edge and cloud service provider, and the way to glue different authentication protocols and message contents between edge and cloud service provider. The authentication protocol in cloud is deployed in an application layer while the authentication protocol in edge (as governed by 3GPP) is in a network layer. This causes a conflict of controlling signals while combining authentication protocols of two sides together. The difference between components and environment also has to be taken into account. As the authentication in edge is handled by the 3GPP cellular network, we have to consider the 3GPP entities that participate in the authentication process, which are the gateways, Mobility Management Entity (MME), and Home Subscriber Server (HSS). On the other hand, an authentication server typically responds to authentication in cloud.

The message content between authentication entities in a 3GPP cellular network is totally different from the authentication message in a cloud which causes a message mismatch. We need to solve such a message mismatch to achieve third party authentication by gluing the two protocols together. The authentication process adopted by OpenID connect (OIDC: authentication protocol in cloud) is simpler than the EPS-AKA protocol (authentication protocol in edge). Thus, we need to design an appropriate protocol through appropriate mapping of messages between cloud and edge.

Solution: Federation Proxy with Four Roles

To solve the issues discussed in the previous section, we propose a proxy-based federated authentication between the cloud and edge. The proxy is divided into two parts in each scenario of a cloud-edge federation where each part communicates with edge side or cloud side. Therefore, the cloud and edge can transparently communicate with the proxy without any modifications. Figures 1c and 1d show the architecture of the proposed solution for cloud-to-edge and edge-to-cloud scenarios. The current cloud service providers do not support SIM cards and we assume that a cloud provider will support the embedded Subscriber Identity Module (eSIM) in the near future. The service that UE desires to access is in a foreign edge deployed by 3GPP cellular network where UE has no account. The authentication, in this case, is based on roaming authentication. Thus, one side of the proxy is virtual HSS and needs to communicate with visited MME (vMME) via protocol s6a. The user communicates with the vMME through evolved Node B (eNodeB).

To obtain the user information from the cloud, the proxy needs to establish a secured tunnel to the cloud by using public-key cryptography. After that, the proxy needs to communicate with the home cloud, so the other part of the proxy will pretend to be a virtual user, which can be seen in Fig. 1c. On the other hand, Fig. 1d represents the architecture of the edge-to-cloud scenario. In this case, SIM can be either eSIM or Universal Integrated Circuit Card (UICC). OIDC is the protocol used among UE, cloud and proxy. Hence, one part of the proxy is an OIDC Provider (OP) which acts as the identity provider in OIDC and is able to connect with the cloud. The other part of the proxy plays the role of a virtual UE, which emulates the typical UE and connects with the 3GPP cellular network because EPS-AKA is used as the authentication protocol between proxy and edge.

Cloud-to-Edge

In the cloud-to-edge scenario, the UE wants to access the services provided by the edge but the account information is in the cloud. The UE will contact the MME of the visited edge which will, in turn, contact the virtual HSS in the proxy as the UE is foreign to the cellular network. The proxy will play the role of a virtual user at the cloud side, obtain the user information from the cloud, give it to the virtual HSS which will return it to the visited MME. In this way, the 3GPP cellular network, which deployed the edge, will think of the whole process as roaming authentication, as it only had to contact with the home HSS of the user. This whole process is divided into three stages where each stage has multiple messages, as shown in the Fig. 2a.

Authentication Information Request Stage (Steps 1 to 4): In this stage, the International Mobile Subscriber identity (IMSI) of the subscriber is sent in the authentication request message through the visited MME to the home HSS, which is a virtual HSS by proxy. The virtual HSS forwards the IMSI to the virtual user part in the proxy by an authentication request message. This sends out another authentication request to the subscriber server in home cloud, based on a normal web login session. A mapping table is designed in the proxy that maps IMSI to ID and password (stored in the proxy) and stores other necessary parameters to communicate with both sides. Here, the home cloud verifies the IMSI. If the IMSI matches with IMSI in user data base, the cloud replies to the virtual user.

Authentication Info Response Stage (Step 5 to Step 8): With the user's information, cloud sends claims containing an authentication vector (AV) to a virtual user. In this case, we assume that the home cloud provides eSIM and thus, the home cloud stores the challenge to verify eSIM. The challenge claims include AUTN || RAND, and XRES. Once the virtual user receives the AV, it forwards it to virtual HSS in proxy. From this step, the authentication is the same as EPS-AKA. Virtual HSS sends AUTN || RAND, and XRES to the visited MME, which retains the XRES and then sends out the AUTN || RAND to UE. Here, eSIM verifies the AUTN for authentication, and then calculates the RES with RAND.

Authentication Confirmation Stage (Step 9 to Step 12): After stage 2, UE forwards the RES in an authentication response to the visited MME which compares RES and XRES to authenticate UE. One confirmation message (200 OK) is sent back to UE. After this step, the mutual authentication between UE and visited 3GPP cellular network (which deploys the edge) is complete. However, the cloud also needs to know that the UE has been authenticated. Therefore, proxy sends an update location message to the home cloud. This message announces not only the UE authentication but also the new MME that the UE is being attached to. After this stage, the cloud and the UE complete the mutual authentication.

Edge-to-Cloud

In an edge-to-cloud scenario, the UE wants to access the services provided by the cloud and has the account information in the HSS of the 3GPP cellular network that has deployed the edge. The UE, having eSIM, will contact the cloud service provider which is the relying party (RP) in this case. This cloud service provider will contact the Identity provider (IdP) in the proxy as per OIDC, which is also named as OIDC provider (OP). The proxy will play the role of Identity provider at the cloud side, and the virtual UE at the edge side. The proxy will pretend to be the UE and will connect with the HSS through MME and obtain the user authentication information and will provide it to the identity provider which will, in turn, pass it to the cloud service provider. In this way, the whole process is transparent for both the cloud and the edge. This process is divided into three stages, where each stage has multiple messages, as shown in Fig. 2b.

Authentication information request stage (Step 1 to Step 5): Initially, the UE sends the service request to the service provider which provides the identity service to the UE. Here, the UE chooses the third-party authentication service supplied by a mobile network provider. The IMSI in USIM is sent as a kind of user ID to OP. After that, proxy with a virtual UE forwards IMSI to the home core network as an authentication request.

Authentication Info Response Stage (Step 6 to Step 12): After receiving the authentication request, the core network sends back an authentication response, which contains RAND || AUTN and XRES. MME stores the XRES, and forwards the challenge RAND || AUTN to the proxy. The proxy continues forwarding the challenge to the UE. After verifying the AUTN, the UE computes RES and sends it back to the proxy. Consequently, the proxy forwards RES to the MME which compares the RES with XRES to authenticate the UE.

Authentication Confirmation Stage (Step 13 to Step 16): After the MME confirms that RES equals XRES, the MME sends an OK message to the proxy. The proxy supplies an authentication token to the UE, which sends it to the service provider. The service provider then authenticates the user by validating an authentication token with OP. If the authentication token is correct, the service provider permits the user to access the service.

The proxy is the vulnerable point of the proposed third-party authentication in both the cases. An attacker can hack it to steal information, hijack the proxy, eavesdrop, or influence the traffic negatively. Therefore, the proxy has an anomaly detector which extracts the log file in the proxy to ascertain if there are any abnormal activities. If a threat is detected, the detector will send an alarm to a threat handler that decides a suitable solution for the proxy. The mapping table stored in the proxy can be secured by using hash function. If attacker somehow obtains the IMSI, it will not be able to extract the correct ID and password of UE.

Apart from the proxy, threats also occur in the form of malicious UE. If UE is malicious, it may send spoofed requests and intercept a challenge at Steps 1 and 8 of the cloud-to-edge scenario, which is shown in Fig. 2a. A spoofed request can be detected by the home cloud through the user's IMSI at Step 4 of Fig. 2a. The challenge interception will also be detected at Step 9 of Fig. 2a, as a malicious UE will not be able to generate the correct RES. In the case of an edge-to-cloud scenario, shown in Fig. 2b, a malicious UE can send spoofed requests, intercept challenges, and intercept tokens at Steps 2, 9, and 15, respectively. In this scenario, the issues of spoofed requests and challenge interception can be handled just like the cloud-to-edge scenario and, as in the case of token interception, the OP in proxy will not validate the authentication code at Step 16 of Fig. 2b. UE can also cause denial of service (DoS) which can make service unavailable and it is a fatal threat for commercial networks. To prevent DoS attacks, the number of UE connecting with eNodeB should be restricted.

Implementation and Results

We implemented the federation proxy solution for cloud-to-edge and edge-to-cloud on the testbed which is shown in Fig. 3, using four computers and one router. One computer was used as

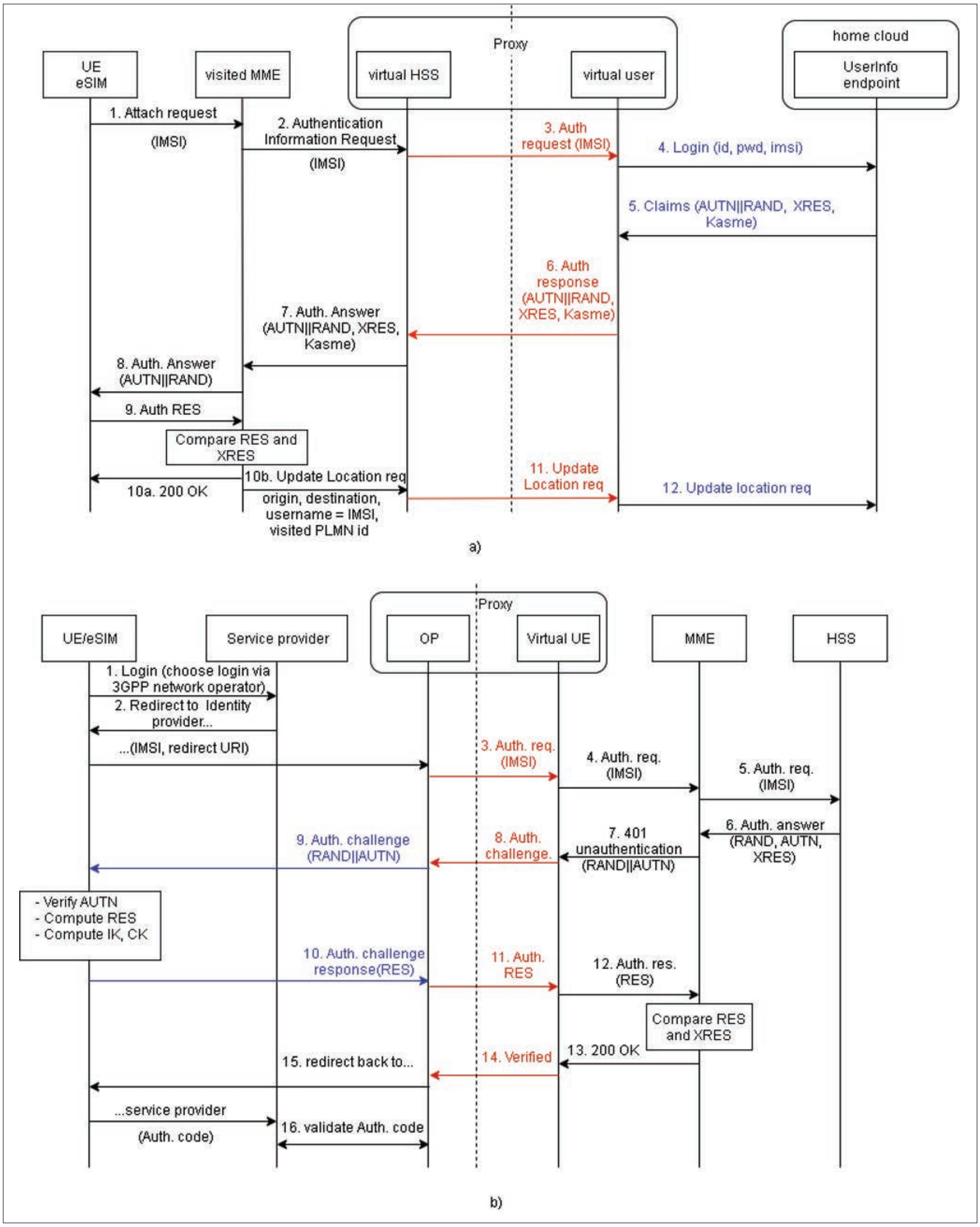

FIGURE 2. Third-party authentication message flow: a) Cloud-to-Edge scenario; b) Edge-to-Cloud scenario.

UE that contained provisioning information as in a SIM card. It also had User Agent or browser to connect with RP. The second computer was configured as a cloud, which contained OP and RP. The third computer included MME and HSS and the fourth one was the proxy. The proxy computer had four parts, namely virtual HSS, virtual user, virtual UE, and virtual OP. These computers all had intel corei7-7700 processors with 16GB RAM and Ubuntu 16.04 OS. The black line in Fig. 3 illustrates cloud-to-edge connection. The path of cloud-to-edge connection included UE, MME, virtual HSS, RP, and OP. On the other hand, the red line in Fig. 3 represents the connection of edge-to-cloud. The UE, router, cloud, virtual OP, virtual UE, MME, and HSS are the elements of this path. The cloud-to-edge and edge-to-cloud scenarios were implemented on this experimental testbed through the following series of steps: creating a web app by using an OIDC framework; installing a UE on a laptop using CryptoMobile source code; creating an identity provider using the OIDC framework on PC 1; emulating MME and HSS on PC 2; emulating a proxy on PC 3 through virtualization of UE, HSS, and OIDC provider. After implementation, we investigated the following issues: the first was the third-party authentication time to find out the difference between the authentication time taken by our proxy-based federated authentication and the authentication time taken if a user had to authenticate with both parties separately. We identified a bottleneck in the approaches in two scenarios. And lastly, we investigated how we could reduce the delay time of these methods to achieve seamless service.

First, we compared the message overhead and delay time between third-party authentication and two-party authentication. We compared the number of messages exchanged in two proposed scenarios and two normal authentications in edge and in cloud, respectively. The number of messages exchanged for authentication protocols in edge-to-cloud scenario was 16, in cloud-to-edge scenario is 12, in OIDC is 9, and in EPS-AKA 5. The total number of messages exchanged for OIDC and EPS-AKA was two messages more than in the cloud-to-edge authentication protocol and two messages less than in the authentication process in edge-to-cloud. This was reasonable because the four messages exchanged in proxy were also counted. Figure 4a compares the delay time between the two proposed scenarios and two normal authentications in edge and in cloud in respect of the authentication stages which were mentioned in the previous section. The graphs show that the third-party authentications in edge-to-cloud and cloud-to-edge were reduced 27.7 percent and 37.9 percent of total authentication time as compared to the concatenation of OIDC and EPS-AKA. The proxy-based federated authentication was compared with the concatenation of OIDC and EPS-AKA because, if the UE application requires services from both the cloud and edge, then UE has to do authentication with both of them, which would be a concatenation of OIDC and EPS-AKA. It can also be seen that authentication time of edge-to-cloud was 16.5 percent longer than cloud-to-edge, because of the four additional messages exchanged in edge-to-cloud than cloud-to-edge protocol.

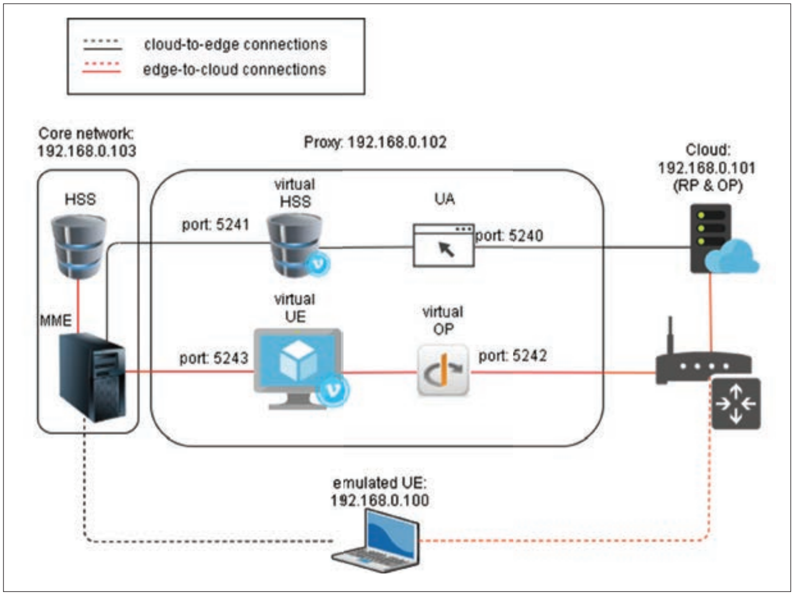
cloud-to-edge connections
edge-to-cloud connections
Core network: 192.168.0.103
Proxy: 192.168.0.102
Cloud: 192.168.0.101 (RP & OP)
HSS
virtual HSS
UA
port: 5241
port: 5240
MME
virtual UE
virtual OP
port: 5243
port: 5242
emulated UE: 192.168.0.100

FIGURE 3. Experimental testbed.

The graphs also show that the two stages, namely authentication information request stage and authentication confirm stage in edge-to-cloud and cloud-to-edge, take similar time as their counterparts in OIDC and EPS-AKA because of the similarity in the exchanged messages. Time taken by the authentication information response stage in edge-to-cloud and cloud-to-edge scenarios is less than the sum of EPS-AKA and OIDC's delay times, because of the use of a wire link instead of wireless link in some steps. It is also one of the reasons why total delay time of edge-to-cloud and cloud-to-edge was less than the combination of OIDC and EPS-AKA.

We also compared the edge-to-cloud and cloud-to-edge scenarios from the perspective of time taken by the protocols involved. Figure 4b shows that most of the delay time of edge-to-cloud authentication protocol was due to the steps in cloud side, while in cloud-to-edge scenario, steps in edge side used a larger proportion of authentication time. The reason is that the message flow of these scenarios and the air interface is further explained in Fig. 5. However, it can be seen that the total time taken by the steps in the cloud in the edge-to-cloud scenario is more than with conventional OIDC, because two messages, an authentication challenge and an authentication challenge response, which contain RAND||AUTN and RES respectively, are modified messages. These take more time for transmission in a wireless environment which increases the time taken by the steps in cloud side for the edge-to-cloud scenario. However, all the steps in the edge side for edge-to-cloud scenario were in the wired environment, which is why the delay time of steps in edge for edge-to-cloud scenario was much smaller than the EPS-AKA. Cloud-to-edge scenario, on the other hand, had delay time in edge slightly longer than conventional EPS-AKA because some messages were added in wired environment. The delay time in proxy is small when compared with overall delay time in both third-party authentications. Delay times among proxy only cover 1.5 percent and 1.6 percent of

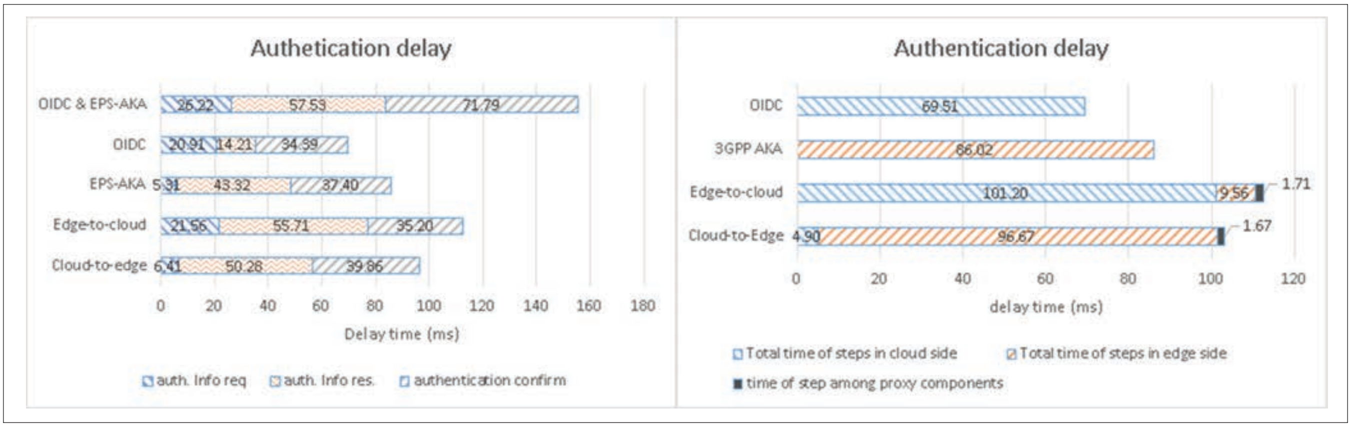

FIGURE 4. Authentication delay: a) based on three stages of authentication protocols; b) based on type of protocol.

total delay time in edge-to-cloud and cloud-to-edge, respectively. Therefore, proxy does not significantly affect on the whole delay time.

The bottleneck in both scenarios was caused by the air interface. Figure 5 shows the proportion of time in which UE was involved (UE uses the air interface to communicate) in exchanging messages. Figures 5a, 5b, 5c, and 5d show the delay times for each step of EPS-AKA, OIDC, cloud-to-edge, and edge-to-cloud, respectively. The blue bars indicate those steps involving air interface. It can be clearly seen that delay time on air interface of all these protocols was much higher than non-air interface. In cloud-to-edge authentication time, the time UE participates in the exchange of messages is 76 percent of the total time. In other words, the delay time of air interface covered over three quarters of overall authentication time of the cloud-to-edge scenario. Similarly, the time UE participated in exchange of messages in edge-to-cloud authentication was even higher and constituted the 87 percent of the total time. It is clear that the messages involving air interface were a clear bottleneck in both scenarios.

In air interface steps, the authentication response step took the largest proportion of the total delay time. For EPS-AKA, cloud-to-edge, and edge-to-cloud, the delay time of authentication response step was more than 42 milliseconds. Because the latency of authentication response was much higher (longer) than for other steps, it was the bottleneck of not only EPS-AKA, but also of the cloud-to-edge and edge-to-cloud. The delay time in OIDC was distributed more evenly than with the three other protocols. It can be inferred that cloud-to-edge and edge-to-cloud federated authentication protocols inherited the bottleneck from the authentication response step of the standard EPS-AKA and the delay time of these proposed protocols can be further reduced if the time taken by the authentication response step of EPS-AKA is reduced.

Conclusions and Future Work

Cloud and Edge computing paradigms provide computational and storage capabilities but neither one of them is good for heterogeneous IoT devices alone. A cloud-edge federation provides the best of both computing paradigms to IoT devices. A user can use services from different providers while only needing one account on one of the providers in a federation. One of the most important challenges faced in federation is third-party authentication, where users with a subscription to one provider need to access service of another provider. This third-party authentication is challenging because the protocols in edge and cloud computing are different from each other and there is a need to bind them together. In this work, we proposed federated third-party authentication by binding existing authentication protocols in edge and cloud, such as EPS-AKA and OIDC. We designed third-party authentication protocols for two scenarios, cloud-to-edge and edge-to-cloud, depending on the user subscription to cloud and edge, respectively. We proposed the deployment of a proxy to bind these protocols together, and this had four roles, two roles for each scenario. We implemented these proposed protocols on a testbed and results show that the third-party authentications in edge-to-cloud and cloud-to-edge respectively reduced 27.7 percent and 37.9 percent of total authentication time as compared to the sum of EPS-AKA and OIDC. Also, with third-party authentication, we do not need to have two accounts on edge as well as cloud. The latencies of third-party authentication in both cloud-to-edge and edge-to-cloud scenario are significantly higher than the delay time of either EPS-AKA or OIDC because of the air interface. In other words, delay on air interface is the bottleneck of the proposed third-party authentication. As 5G technology includes many new wireless technologies, such as massive MIMO, the delay time on air interface will not be the major problem in near future. This third-party authentication in the cloud-edge federation proposed in this article can also be extended to fourth-party authentication, such as cloud-edge-edge, cloud-cloud-edge and so on.

References

[1] A. Alelaiwi, "An Efficient Method of Computation Offloading in an Edge Cloud Platform," *J. Parallel and Distributed Computing*, vol. 127, 2019, pp. 58–64.

[2] ETSI, "Network Function Virtualisation (NFV); Management and Orchestration; Architectural Option," *European Telecommunication Standard Institute*, *GS NFV-IFA 009*, July 2016.

[3] Shareef *et al.*, "A Survey on Federation Cloud Environment," *Int'l J. Advanced Research in Computer Science and Software Engineering*, vol. 5, no. 2, Feb. 2015, pp. 83–92.

[4] R. Roman, J. Lopez, and M. Mambo, "Mobile Edge Computing: A Survey and Analysis of Security Threats and Challenges," *Future Generation Computer Systems*, vol. 78, 2018, pp. 680–98.

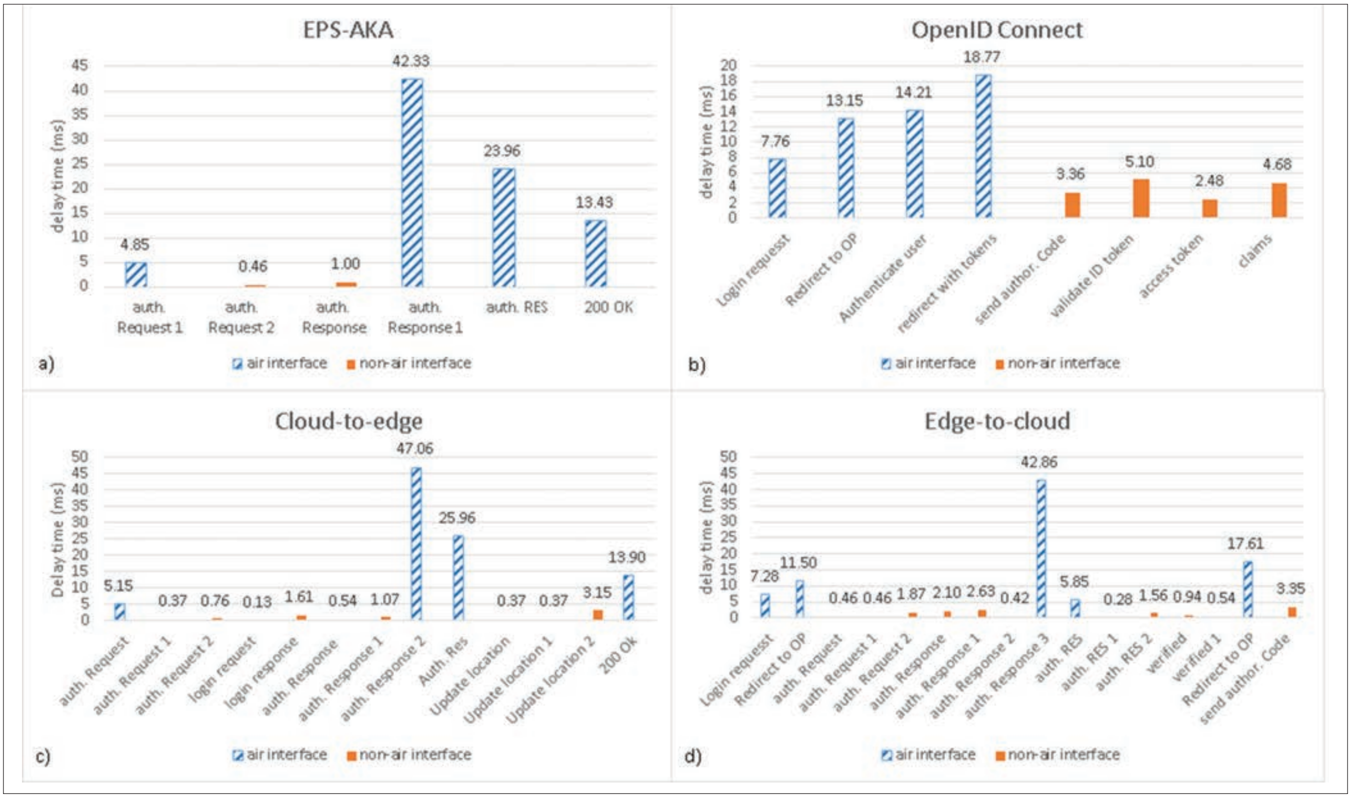

FIGURE 5. Delay on air interface vs. non-air interface.

[5] M. Leandro *et al.*, "Multi-Tenancy Authorization System with Federated Identity for Cloud-Based Environments Using Shibboleth," *Proc. 11th Int'l. Conf. Networks*, 2012.

[6] A. Celesti *et al.*, "Three-Phase Cross-Cloud Federation Model: The Cloud SSO Authentication," *Proc. 2010 2nd Int'l. Conf. Advances in Future Internet (AFIN)*, IEEE, 2010.

[7] Z. Ahmad, J.-L. Ab Manan, and S. Sulaiman, "User Requirement Model for Federated Identities Threats," *Proc. 2010 3rd Int'l. Conf. Advanced Computer theory and Engineering (ICACTE)*, vol. 6, IEEE, 2010.

[8] M. Gogna and C. R. Krishna, "Cross-Domain Authentication and Interoperability Scheme for Federated Cloud," *Smart Systems and IoT: Innovations in Computing*, Springer, Singapore, 2020, pp. 451–61.

[9] A. Alelaiwi, "An Efficient Method of Computation Offloading in an Edge Cloud Platform," *J. Parallel and Distributed Computing*, vol. 127, 2019, pp. 58–64.

[10] R. Wantanabe and T. Tanaka, "Federated Authentication Mechanism Using Cellular Phone — Collaboration with OpenID," *Proc. Int'l. Conf. Information Technology: New Generations*, 2009.

[11] I. Friese et al., "Bridging IMS and Internet Identity," *Proc. 2010 14th Int'l. Conf. Intelligence in Next Generation Networks (ICIN)*.

[12] A. Sharaga and A. Luft, "Multi-Hop Single Sign-On (SSO) for Identity Provider (IDP) Roaming/Proxy," U.S. Patent 9,258,344 B2, Feb. 9, 2016.

[13] K. Gibbons, J. O'Raw, and K. Curran "Security Evaluation of the OAuth 2.0 Framework," *Information Management and Computer Security*, vol. 22, no. 3, Dec. 2014.

[14] F. Lordan, J. Jensen, and R. M. Badia, "Towards Mobile Cloud Computing with Single Sign-on Access," *J. Grid Computing*, vol. 16, 2018, pp. 627–46.

[15] R. Khan, J. Ylitalo, and A. S. Ahmed, "OpenID Authentication as a Service in OpenStack," *Proc. 7th Int'l. Conf. Information Assurance and Security*, IAS 2011, Melacca, Malaysia, Dec. 5–8, 2011

Biographies

Ying-Dar Lin is a distinguished professor of computer science at National Chiao Tung University (NCTU), Taiwan. He received his Ph.D. in computer science from the University of California at Los Angeles (UCLA) in 1993. Since 2002, he has been the founder and director of the Network Benchmarking Lab. He has served or is serving on the editorial boards of several IEEE journals and magazines, and is the Editor-in-Chief of *IEEE Communications Surveys and Tutorials* (COMST). He has published a textbook, *Computer Networks: An Open Source Approach*. His research interests include network security, wireless communications, and network softwarization.

Truong Duc-Tai received his bachelor degree in electrical and telecommunication from the University of Engineering and Technology, Vietnam National University, Hanoi. Currently, he is a senior masters student of computer science at National Chiao-Tung University, Taiwan. He is interested in wireless communication, IoT, and network security.

Asad Ali received his master degree in electrical engineering from the National University of Science & Technology, Pakistan. Currently, he is pursuing his Ph.D. degree from National Chiao Tung University, Taiwan. His research interests are network security, wireless communications, network design and optimization.

Chi-Yu Li is an assistant professor in the Department of Computer Science at NCTU and leads the NEtworking and Mobile Systems (NEMS) laboratory. He received his Ph.D. degree in computer science from UCLA. He received his M.S. degree in computer science from NCTU in 2006, and two bachelor's degrees in computer science and management science from NCTU in 2004. His research interests include wireless networking, mobile networks and systems, and network security.

Yuan-Cheng Lai received his Ph.D. degree from the Department of Computer and Information Science from National Chiao Tung University in 1997. He joined the faculty of the Department of Information Management at National Taiwan University of Science and Technology in August 2001 and has been a professor since February 2008. His research interests include performance analysis, network security, 5G mobile networks and Internet of Things.

Thai-Mai Thi Dinh is an assistant professor on the Faculty of Electronics and Telecommunications, VNU University of Engineering and Technology, Hanoi, Vietnam. She graduated from the Post and Telecommunication Institute of Technology, Vietnam in 2006. She received the master and Ph.D. degrees from Paris Sud 11, France in 2008 and VNU University of Engineering and Technology, Hanoi, Vietnam in 2016, respectively. Her research interests focus on mobile networks, wireless communications and indoor positioning systems.