A Strong Authentication Method for Web/Mobile Services

Samy Kambou, Ahmed Bouabdallah

A Strong Authentication Method for Web/Mobile Services

Samy Kambou & Ahmed Bouabdallah
IMT Atlantique
IRISA, UBL
F-35576 Cesson Sévigné, France
{samy.kambou, ahmed.bouabdallah}@imt-atlantique.fr

***Abstract*—This paper presents an original and strong authentication method for cloud services from smartphones. It is based on a two-factor scheme improved by the diversity of devices and network channels. It combines an OTP[1]-based approach using an IoT[2] object as a secondary device. Authentication factors are transmitted over different channels implementing distinct protocol stacks (LTE[3], LPWAN[4], ...). The proposal uses end-to-end encryption for the transfer of sensitive data. An experimental application of the method is illustrated by analyzing authorization access issues to cloud services located in a trusted zone. A platform developed to test the approach is briefly presented.**

***Keywords*-strong authentication, multi-factor authentication, internet of things, one-time password, diversity, cloud-based web service**

I. Introduction

The smartphone has gradually become the indispensable companion for all daily tasks. Its mobility capabilities, combined with its growing processing power and autonomy, allow end users to access most online cloud services from anywhere at any time. Such a change of use thus leads the smartphone to gradually move from the position of secondary device to that of primary device (formerly occupied by the laptop computer).

Such an evolution has many impacts including those related to two-factor authentication methods for cloud services from a computer and involving the smartphone as a second terminal. In this case, the first step of authentication takes place between a service located in the cloud and the end user connected from a computer. If successful, the service sends a specific code by SMS to the end user's smartphone. The latter relays it to the service via his computer. The service validates this second step if the code received is the same as the one sent initially. The multiplicity of devices and communication channels of such a scheme reduces its attack surface and is at the origin of its robustness. But this one will be clearly lost if applied with a smartphone used both as a primary and secondary device.

This work has received funding from the France Brittany region research and innovation program AAP PME 2016, under grant agreement number 16004254, V2OLTERES project.

[1]OTP: One-Time Password
[2]IoT: Internet of Things
[3]LTE: Long Term Evolution
[4]LPWAN: Long Range Radio Wide Area Network

It is however possible to keep the principle of this scheme and at the same time maintain its robustness by introducing a smart-Thing [1] as secondary device. We build on this approach to design and develop an authentication protocol that provides a high level of assurance. In the proposed solution the end user submits with his mobile phone the first authentication factor and then directly access a cloud service. The second factor which is OTP-based [2], is exchanged between the smart-Thing and the service through a protocol of the LPWAN family [3]. To the best of our knowledge, there have been no other authentication scheme proposed in the literature using an LPWAN network as an alternative channel to provide two-factor authentication from a mobile to cloud-based web services.

The paper begins with some background about authentication schemes and quickly sketches the IoT environment. The proposed authentication method is then described in detail. To illustrate the approach, a common use case of authentication is then conducted by analyzing authorization access issues to cloud services. A platform developed to test the approach is briefly sketched.

II. Related works

A. Authentication methods

Authenticating an information consists in verifying its authenticity. In our case it consists in being able to remotely attest the authenticity of a claimed identity. Almost all existing approaches involve checking a predefined link [4] between the targeted identity and other dedicated information that we will call *credential*. Credentials are usually classified as :

- "*something you know*" which can be seen as a *knowledge credential* (*KC*). We may quote for example textual or graphical passwords, personal identification numbers (PIN), lock patterns etc.
- "*something you have*" which falls in the *possession credential* (*PC*) class. We can mention smart cards, hardware tokens etc.
- "*something you are*" which belongs to the *inheritance credential* (*IC*) category. We can cite mainly physiological features as fingerprint, iris, voice, DNA, etc.

This list is far from being exhaustive because it is indeed possible to use other kind of credentials (for example "where

you are" like geographical location, IP address, ... or more generally contextual information).

The intensity of attacks against authentication methods has necessitated their reinforcement by the verification of additional credentials in the framework of multi-factor methods [5]. Several multi-factor authentication systems have been proposed for a wide variety of purposes. Some of the most recently proposed mobile-based authentication are as follows.

TDAS [6] is a touch dynamics based multi-factor authentication system for mobile devices. The proposed approach aims to study the feasibility and benefits of adopting an authentication method based on touch dynamics mechanism (*IC*) by integrating it with the PIN-based authentication method (*KC*). The authors presented how the data set may be used to strengthen the protection of resources that are accessible on mobile devices.

Another approach is by Crossman and Liu [7], who propose a two-factor authentication based on NFC smart-phone devices. Firstly, users are asked to enter a password (*KC*) which unlocks the protection of the key on their mobile phones. Then, the key (*PC*) is transferred through NFC to complete the authentication process. In this system, the two factors are managed by the same mobile phone. This assumes a central point of vulnerability that can potentially be used by an attacker. Indeed, if the attacker compromises the mobile phone, the two authentication factors will be easily available.

Barkadehi et al. [8] proposed another two-factor authentication system by using the mobile phone as a mirror. In their proposed system, a web application uses a username/password as an authentication factor (*KC*) in the first step and then a white box will be shown to the users. The users cannot see the mouse cursor in the box but must move their mouse in the box in order to click the right second password. At the same time, they receive a notification on their mobile phone (*PC*) to open the mirror application. Then, they must accept the received request to continue the authentication process and they will see their web-based cursor on a shuffled keyboard in their mobile application. To conclude the authentication process, the users need to select their second password. After a valid authentication process, the users will have access to a web service through their laptop. Our scheme involves the use of a mobile phone to access a cloud-based web service. Another smart device is integrated to improve the security.

Our solution is also based on OTP-based two-factor authentication and is divided into two main steps:

- the mobile phone will be used to control the first factor.
- the smart device will have to manage the second factor.

B. IoT environment

With rapid development of Internet services and wireless technologies, intelligent mobile devices are well integrated into our daily life to provide customized Internet of things (IoT) to individuals. Every intelligent mobile object is now capable of interconnection, storing data and receiving user command to accomplish tasks requested by users. Heterogeneous system architectures are formed in which different types of devices as well as the relevant communication techniques are deployed. Conventional security mechanisms must therefore be refined to fit the requirements from IoT environment.

An IoT device is integrated into our OTP-based two-factor authentication scheme. This smart-Thing must implement a network stacl of the LPWAN family [3] such as LoRa/SigFox networks or one of the latest 3GPP technologies dedicated to the IoT solutions which are LTE cat m1 and NB-IoT networks. The IoT device must also be an active object meaning that it must have its own power supply and be able to perform some cryptographic operations.

III. Strong Mobile authentication Scheme for Web/Mobile Applications

Strong authentication is a technique relying on more than one authentication credential. By combining *something you know* and *something you have*, an adversary needs to physically steal your hardware token and also learn your password. This two-factor authentication scheme provides improved protection. In addition communication channels' and devices' diversity is another way to further improve it. By this way, attacks such as man in the middle or eavesdropping are much more difficult to carry out since the adversary must simultaneously control both channels and devices. We introduce below a strong authentication method on mobile using an IoT device and exploiting these two improvements.

To authenticate a user on his smartphone and using an IoT device as a security token, certain components must be in place. Fig. 1 shows the basic architecture and the main components used to design the proposed scheme.

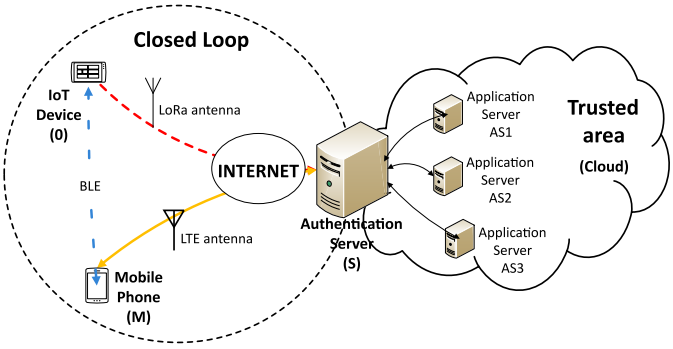

Fig. 1. Architecture of mobile authentication using an IoT device

The user must have a smartphone (M) connected to Internet through LTE (or WIFI) and also be in possession of an IoT device (O). These two devices must be equipped with BLE[5] (or NFC[6]) technology that is increasingly be leveraged to create more mobile-friendly experiences. By using his mobile, the user can access to web services located in the Cloud subject to prior authentication. These web services hosted by servers disseminated in the Cloud are connected to S which is the

[5]BLE: Bluetooth Low Energy
[6]NFC: Near Field Communication

front-end that will manage the authentication process. S is also connected to a LPWAN network through Internet allowing in this way its communication with the IoT device O.

We propose a strong authentication scheme involving two distinct user's devices which communicate over different networks. One important point concerns the fact that both devices are controlled by the same user. This is ensured by a *closed loop* going through all the components involved in our architecture illustrated in Fig. 1. The loop starts in the mobile requesting the service, goes through the network with S and then via the IoT device and back to the mobile. This closed loop can be realized in several ways. Below, we describe how to use it in our strong authentication scheme.

IV. Authentication protocol in detail

We first summarize in the following table, all the notations used in the descriptions below.

TABLE I
NOTATIONS

u_{id} / pwd	User identifier / User password
M	Mobile phone
O	IoT device
S	Authentication server
id_m / id_s	Mobile phone identifier / Server identifier
t_1 / t_2	Authentication token / Access token
sec_{OTP} / cod_{OTP}	OTP secret / OTP code
K_*^{priv}, K_*^{pub}	Private-Public key
h_1, h_2, h_3	Hash functions
$E_{sec}(*)$	Encrypt function with the secret sec
$Sg_{see}(*)$ / $XSg_{see}(*)$	Signature/Verification with the seed see

The protocol rests on several security services. We first describe the main steps of the protocol while abstracting security issues. We then provide a detailed account of the required underlying security services.

A. Protocol description

The proposed method rests on three distinct channels:

- a *primary* channel between the mobile phone M and the authentication server S
- a *secondary* channel between S and the IoT device O
- a *tertiary* channel between O and M

In the developed prototype, we instantiate these three categories respectively with LTE, LoRa and BLE. There are therefore many other choices that must however take into account the technical constraints imposed by the devices. These channels require at least confidentiality and integrity for exchanged messages. To avoid depending on the protocol security specificities used to instantiate a channel, we use a security layer detailed in section IV-B to independently guarantee the requirements for each communication channel.

We now focus on the flow of information required for an authentication as outlined in Fig. 2, the steps of which are detailed below:

1) By using his mobile phone M, the user submits a username (u_{id}) and password (pwd) to the authentication server S. These credentials (F1) are transmitted over an LTE link. This is the initial step of the authentication protocol during which the first *something you know* factor is used.
$$M \rightarrow S : [u_{id}, pwd]^{LTE}$$

2) S verifies the received data. If the data match with those stored in the database (DB), S replies with an authentication token (t_1) and an OTP secret (sec_{OTP}). t_1 is represented as a JWT (JSON Web Token) containing at least the u_{id} and works as a session identifier. sec_{OTP} is a random number generated at each authentication request. It is used as a seed in step 4 below.
$$S : u_{id}; pwd =? \ DB\{u_{id}; pwd\}$$
$$S \rightarrow M : [t_1, sec_{OTP}]^{LTE}$$

3) M verifies that S generated the OTP secret sec_{OTP}. Both received data are sent to O through a BLE link.
$$M : Generator(sec_{OTP}) =? \ S$$
$$M \rightarrow O : [t_1, sec_{OTP}]^{BLE}$$

4) O generates an OTP code (cod_{OTP}) by applying a function ($func$) involving sec_{OTP}. cod_{OTP} has a predefined validity period and is associated with t_1 to build a new credential (F2) needed to the second step of the authentication protocol. This second *something you have* factor is sent to S via a LoRa network.
$$O : cod_{OTP} = func_{OTP}(sec_{OTP})$$
$$O \rightarrow S : [t_1, cod_{OTP}]^{LoRa}$$

5) S computes its own version of the OTP code ($Xcod_{OTP}$) and compares it to the one contained in the credential (F2). If the data match, S validates the authentication. In anticipation of access authorization aspects analyzed in section V-A, S replies with an access token (t_2). We will explain in the section V-A how t_2 can be used to access to resources or services.
$$S : cod_{OTP} =? \ Xcod_{OTP}$$
$$S \rightarrow O : [t_2]^{LoRa}$$

6) O forwards the access token to M via the BLE link.
$$O \rightarrow M : [t_2]^{BLE}$$

B. Security enforcement of the protocol

Security brings a panoply of challenges in the authentication protocol design. It is therefore very difficult to build a secure authentication protocol [9]. Formal proofs could provide guarantees on the correctness of the protocol. However failing to provide a formal proof which at the time of writing this paper is still in progress, we introduce below the sound principles followed in the design of this protocol.

Firstly, OTP is introduced in our proposal as the second authentication factor. The OTP codes are generated by the IoT device and can therefore appear totally random. As the name suggests, each OTP code is only used once. By changing each time an OTP code is needed, OTP solution introduces liveness. This is an important issue in security.

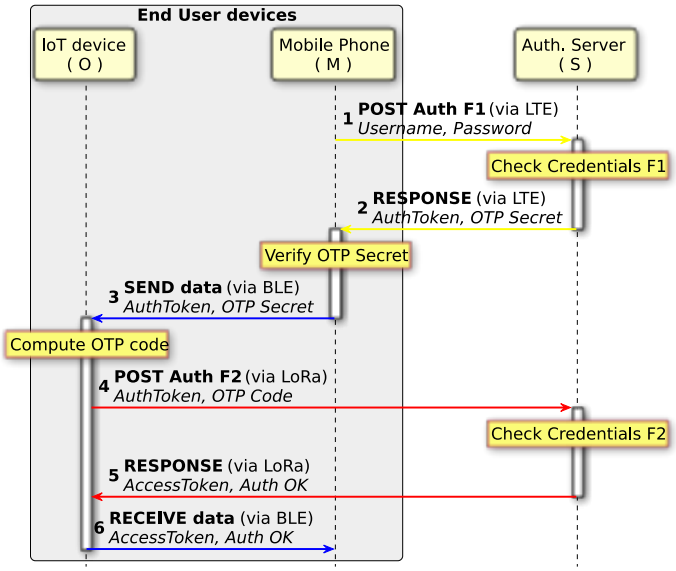

Fig. 2. Data flow of Authentication Procedure

Secondly, we took advantage of JWT [10] which are tokens using a container to transport data between interested parties in JSON. These tokens are base64-encoded. We use them to authenticate all exchanged requests in a RESTFUL approach. They are defined with an expiration time that avoids forever valid tokens. The tokens are also protected against replay attacks by applying timestamps.

Thirdly, the three communication channels are secured with encryption algorithms.

1) *LTE Channel: Mobile Phone ↔ Authentication Server*

- *Key Agreement*: Before any data transmission, ECDH(E) (Elliptic Curve DiffieHellman, where final "E" stands for "ephemeral") and ECDSA (Elliptic Curve Digital Signature Algorithm) algorithms are used to create a secure channel between each mobile phone and the authentication server. These algorithms allow to define a trusted session key (ms). ECDH [11] is a well-known key agreement protocol used to define a shared secret over an insecure channel. Each involved party must have an elliptic curve public-private key pair. ECDH(E) is a variant of ECDH which provides temporary key pair instead of trusted static key pairs (via a certificate). ECDSA [12] is a variant of Digital Signature Algorithm (DSA) which uses the elliptic curve cryptography. This algorithm is used by a signatory to affix a *digital signature* on data and by a checker to prove the validity of the signature. Each party involved has a public-private key pair. The private key operates in the signature generation process and the public key is used for the signature verification. ECDSA provides message authentication, integrity and non-repudiation.

  a) M generates its keys pair (K_m^{priv}, K_m^{pub}) and sends its public key and its signature to S.
  $M : K_m^{priv}, K_m^{pub}$
  $M : Sg_{K_m^{pub}}(h_1(id_m))$
  $M \rightarrow S : [K_m^{pub}, Sg_{K_m^{pub}}(h_1(id_m))]^{LTE}$

  b) S verifies the received signature to ensure that the data come from M (we make the assumption that S has previously registered the respective identities of all the legitimate devices which can be used for a user authentication). If the data come from a known device, S generates its pair of keys (K_s^{priv}, K_s^{pub}). Then, it computes the session key (ms) using its private key and the public key of M. Finally, S replies to M with its public key and its signature.
  $S : XSg_{K_m^{pub}}(h_1(id_m)) = ?Sg_{K_m^{pub}}(h_1(id_m))$
  $S : K_s^{priv}, K_s^{pub}$
  $S : ms = Compute\ [K_s^{priv}, K_m^{pub}]$
  $S \rightarrow M : [K_s^{pub}, Sg_{K_s^{pub}}(h_1(id_s))]^{LTE}$

  c) M verifies the received signature to ensure that the data come from S (we make the assumption that M has previously registered the identity of the authentication server S). Then M computes the session key (ms) using its private key and the public key of S.
  $M : XSg_{K_s^{pub}}(h_1(id_s)) = ?Sg_{K_s^{pub}}(h_1(id_s))$
  $M : ms = Compute\ [K_m^{priv}, K_s^{pub}]$

- *Symmetric Encryption*: After the key negotiation, the channel is secured with AES cipher [13] (standard recommended by NIST) using the key (ms) to provide end-to-end data encryption and integrity $(E_{ms}(*))$. Furthermore, OTP secret is signed $(Sg_{id_m}(*))$ by the authentication server using the mobile identifier as a seed. Thus, the secure data are completely binded to a specific mobile phone which is by the way ensured that sensitive information received on this channel come from the authentication server. The complete respective expressions of step 1 and 2 introduced in section IV-A are thus:
  $M \rightarrow S : [E_{ms}(u_{id}), E_{ms}(h_2(pwd))]^{LTE}$
  $S \rightarrow M : [E_{ms}(t_1), Sg_{id_M}(E_{ms}(sec_{OTP}))]^{LTE}$

2) *BLE Channel: Mobile Phone ↔ IoT device*
This channel is secured with symmetric AES encryption. More precisely, the AES cipher with 128-bit pre-shared key length (mo) is implemented to provide end-to-end data encryption. The data being transferred over this channel are tokens and OTP secrets which are already secure. Step 3 and step 6 of IV-A are refined below:
$M \rightarrow O : [E_{mo}(E_{ms}(t_1)), E_{mo}(sec_{OTP})]^{BLE}$
$O \rightarrow M : [E_{mo}(E_{ms}(t_2))]^{BLE}$

3) *LoRa Channel: IoT device ↔ Authentication Server*
This link is secured with the AES cipher with 128-bit pre-shared key length (so) to provide end-to-end data encryption. Step 4 and step 5 of IV-A are refined below:

$$O \rightarrow S : [E_{ms}(t_1), E_{so}(cod_{OTP})]^{LoRa}$$
$$S \rightarrow O : [E_{so}(E_{ms}(t_2))]^{LoRa}$$

V. Prototype Implementation

We have set up a test platform to experimentally evaluate the proposed strong authentication method. However, an authentication mechanism is usually associated with an authorization process that allows an authenticated user to access a specific part of an application server. In order to evaluate the proposed authentication method in a common context, we have investigated in the following paragraph a possible integration of our authentication method with an authorization control giving access to a cloud-based web service. We look at the particular case of services located in a trusted area, which deserves attention because it is very common in practice. An extension of this approach relaxing this trust-assumption by integrating our method in a general frameworks such as OpenId Connect and OAuth 2.0, is presented in [14].

A. Authorization control for a web service deployed in a trusted area

In this configuration, the authorization to access to services is also granted by the previous authentication server (S) thanks to the *accessToken* (t_2) obtained after authentication process. The idea is to first limit access to the trusted area before granting or not the protected service, requested by an authenticated user. In the rest of the paper, we consider the authentication and authorization server (AAS) instead of the authentication server (S).

Fig. 3 shows the data flow needed to authorize access to an application server (AS) located in the trusted area. First, the authenticated user initiates a request to access to a protected resource or service using an authorization parameter. This parameter contains the encryption value of the *AccessToken* (t_2) concatenated with a random value ($RAND$). As we have introduced in the previous sections, t_2 is a sensitive data required for any secure request toward the trusted area which can not be observed in plain text. It must pass through the network in encrypted form. To improve the security, $RAND$ is introduced to ensure that previous access requests cannot be reused in replay attacks. AES encryption is used with a 128-bit pre-shared key ($E_{mas}(*)$) length only known by M and AS to secure the link.

$$M : Generate\ RAND$$
$$M \rightarrow AS : [E_{mas}(t_2||RAND)]$$

$RAND$ is also transmitted by M to AAS. It is used to compute a response to the challenge (RES) giving access to a protected resource or service.

$$M \rightarrow AAS : [t_2, RAND]$$
$$AAS : Compute\ RES = h_3(RAND)$$

AS decrypts the received data and computes its own version of the response to the challenge ($XRES$) using $RAND$. Then, AS used t_2 to make a valid request to AAS by sending $XRES$.

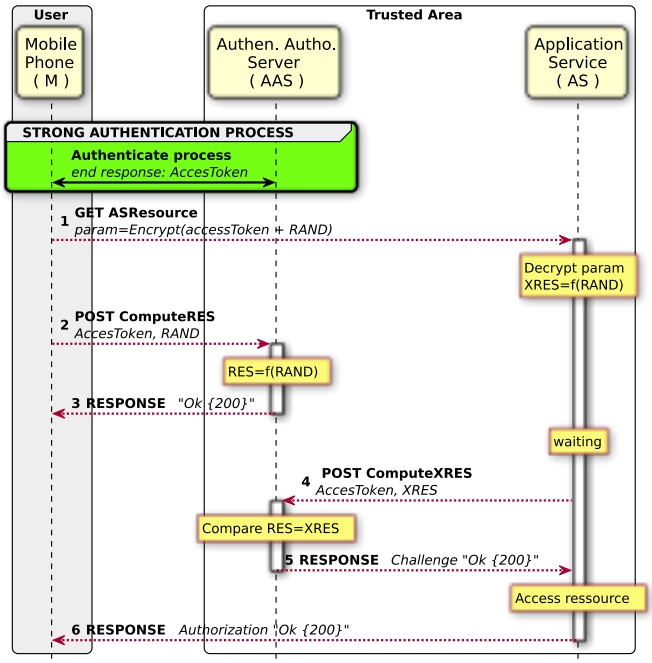

Fig. 3. Authentication and Authorization in a Trusted Area

$$AS : Compute\ XRES = h_3(RAND)$$
$$AS \rightarrow AAS : [t_2, XRES]$$

If AAS answers the challenge correctly (*Challenge "OK"*), the protected resource can be transmitted by AS to M.

$$AAS : RES\ =?\ XRES$$
$$AAS \rightarrow AS : [Challenge\ "OK"]$$
$$AS \rightarrow M : [Authorization\ "OK"]$$

B. Global description of the test platform

Fig. 4 shows two main parts of the architecture. The first is the trusted area in which several servers have been implemented using the open-source cross-platform JavaScript run-time environment *Node JS* [15]. The second part brings together the end user devices.

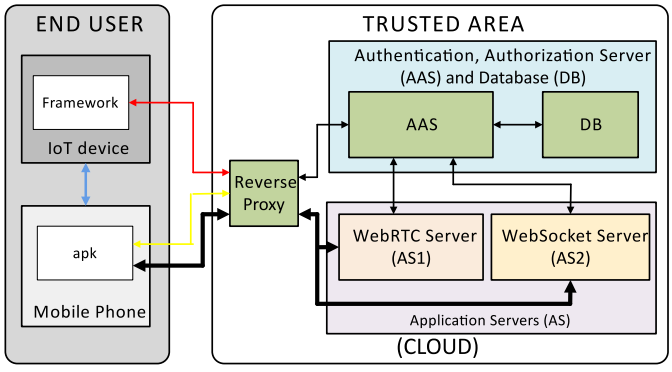

Fig. 4. Platform Architecture

1) Components of the trusted area (cloud): This trusted area is a powerful physical infrastructure using virtualization software to divide it into virtual servers:

a) A Reverse Proxy: it is the entry point to the trusted area. It channels all requests from the Internet and forwards them to the appropriate servers.

b) An Authentication Authorization Server (AAS): It processes an authentication request to define if a previously registered user is really who he claims to be. After successfully authentication, an authorization mechanism determines precisely whether this authenticated user has access to a requested service by evaluating the authorization policy associated to his profile.

c) A DataBase (DB): It collects all user data (user name, password, email, etc) in an organized manner to facilitate access, management and updating. The cross-platform document-oriented MongoDB [16] database has been selected.

d) Applications Servers (AS): For the sake of experiment, we have implemented two different application servers: a *WebRTC server* [17] which provides real-time peer-to-peer audio, video or data (i.e. multimedia) communications by leveraging a set of plugin-free APIs that are available in browsers and a *WebSocket server* [18] which allows full duplex communication over a single TCP connection. It has been used to establish a bi-directional real-time message flow.

2) End User devices: There are two types of device managed by the end user. A smartphone which represents the primary user device and an IoT device which is the secondary user device.

a) User primary device: We used Samsung Galaxy S8 powered with Android Oreo to initiate an authentication process and to perform either a audio/video call or chat in an instant messaging service. Therefore an application (apk) has been developed using Ionic [19] framework. Ionic works with Cordova [20] plugins for the use of some phone resources such as Bluetooth, Camera, GPS, etc. Moreover, Ionic can also creates apk for Android, iOS or Windows systems.

b) User secondary device: We use the Next Generation IoT Platform (Pycom) [21] which provides powerful and affordable MicroPython enabled, multi-network micro-controller development boards. FIPY board [21] is the one we have chosen based on its network communication capabilities: BLE, LoRa, SigFox, LTE Cat-M and NB-IoT.

VI. Conclusion

This paper introduces an original and strong authentication method for cloud services from mobile phones. It combines an OTP-based approach with an IoT object as a secondary device. The proposal implements a multi-factor scheme using channel and device diversity. To avoid depending on the protocol security specificities instantiating a particular channel, we introduce a security layer which provides end-to-end encryption of each involved channel. A platform that implements this method has been developed and tested through the use case of authorization access to cloud services located in a trusted zone.

VII. Acknowledgment

The authors thank the company Acklio which graciously provided them with access to their LoRa network.

References

[1] M. Kuniavsky, "Smart things: ubiquitous computing user experience design," Elsevier, 2010.
[2] M. H. Eldefrawy and K. Alghathbar and M. K. Khan, "OTP-Based Two-Factor Authentication Using Mobile Phones," 2011 Eighth International Conference on Information Technology: New Generations, pp. 327–331, Apr. 2011.
[3] L. Krupka, L. Vojtech and M. Neruda, "The issue of LPWAN technology coexistence in IoT environment," 17th International Conference on Mechatronics - Mechatronika (ME), pp. 1-8, Dec. 2016.
[4] R. Gorrieri and P.F. Syverson, "Varieties of Authentication," Computer Security Foundations Workshop, pp. 79-82, Rockport, MA, June 1998.
[5] D. P. Roberto, M. Gianluigi and S. M. Adriano, "A two-factor mobile authentication scheme for secure financial transactions," IEEE International Conference on Mobile Business (ICMB'05), pp. 28–34, 2005.
[6] P. Shen Teh, N. Zhang, A. Beng Jin Teoh, K. Chen, "TDAS: a touch dynamics based multi-factor authentication solution for mobile devices," International Journal of Pervasive Computing and Communications, Vol. 12 Issue: 1, pp.127–153, Feb. 2016.
[7] M. A. Crossman and H. Liu, "Two-factor authentication through near field communication," IEEE Symposium on Technologies for Homeland Security (HST), pp. 1–5, Mar. 2016.
[8] M.H. Barkadehi, M. Nilashi, O. Ibrahim, "A novel two-factor authentication system robust against shoulder surfing," J. Soft Comput. Decis. Support Syst., Vol.4 Issue: 1, pp. 19–25, Feb. 2017.
[9] M. Abadi and R. Needham, "Prudent engineering practice for cryptographic protocols," IEEE transactions on Software Engineering 22 (1), 6-15, 1996.
[10] M.B. Jones, J. Bradley and N. Sakimura, "JSON Web Token (JWT)", IETF RFC 7519, 2015.
[11] A. Maryam, S. Baharan, A. Difo, R. Amirhossein and O. Habeeb, "Diffie-Hellman and its application in security protocols," International Journal of Engineering Science and Innovative Technology (IJESIT), Vol.1, pp. 69–73, Nov. 2012.
[12] J. Don, M. Alfred and V. Scott, "The elliptic curve digital signature algorithm (ECDSA)," International journal of information security, Vol.1 Issue: 1, pp. 36–63, Jul. 2001.
[13] V. Saicheur and K. Piromsopa, "An implementation of AES-128 and AES-512 on Apple mobile processor," 14th International Conference on Electrical Engineering/Electronics, Computer, Telecommunications and Information Technology (ECTI-CON), pp. 389-392, Nov. 2017.
[14] S. Kambou and A. Bouabdallah, "Using structural diversity to enforce strong authentication of mobiles to the cloud," 5th IEEE Workshop on Security and Privacy in the Cloud, 10-12 June 2019, Washington D.C., USA.
[15] W. Emily, "Nodejs In a Day", CreateSpace Independent Publishing Platform, 2016.
[16] B. Kyle, "MongoDB in action", Manning Publications Co., 2011.
[17] WebRTC Web Site : https://webrtc.org/, (accessed March 31, 2019)
[18] I. Fette and A. Melnikov , "The WebSocket Protocol," IETF RFC 6455, 2011.
[19] Ionic Web Site: https://ionicframework.com/, (accessed March 31, 2019)
[20] Cordova Plugin Web Site: https://cordova.apache.org/, (accessed March 31, 2019)
[21] Pycom Web Site: https://pycom.io/, (accessed March 31, 2019)