Federated IoT Roaming using Private DNS Resolutions

Arnol Lemogue, Ivan Martinez, Laurent Toutain, Ahmed Bouabdallah

Federated IoT Roaming using Private DNS Resolutions

Arnol Lemogue*, Ivan Martinez*, Laurent Toutain*, Ahmed Bouabdallah*
*IRISA, UMR CNRS 6074, F-35700, IMT Atlantique, Rennes, France.

{firstname.surname}@imt-atlantique.fr*

***Abstract*—We are witnessing an exponential increase in IoT applications resting on global mobility of objects. Mobility requires moreover roaming to allow any device to move from one IoT network operator to another. The solutions proposed in the literature are however neither scalable nor secure. The consequences for devices' owners with large fleets of roaming objects may seriously impact the economic benefits of the involved services. We tackle these issues starting from IoTRoam a compliant open-source implementation of the LoRaWAN roaming architecture. This one based on regular DNS resolutions suffers from the limitations indicated above. We propose to modify IoTRoam by introducing a new entity called DNS Broker responsible for orchestrating in a decentralised fashion roaming between different operators using private and secure DNS resolutions. We experimentally show the feasibility of our approach. A preliminary performance evaluation of our implemented architecture shows equal performances for a more flexible, scalable and secure architecture.**

***Index Terms*—IoT, LoRaWAN, Roaming, DNS, DoH, Broker.**

I. Introduction

By connecting everyday objects to networks, the Internet of Things (IoT) is a significant evolution of the digital society. Its exceptional development is reflected in the estimated number of connected objects of around 75 billion by 2025 [1]. IoT has a significant impact on both B2B and B2C services in all sectors of activity linked to ICT (environment, agriculture, health, etc.). Some of the usages imply moving devices.

Low-power Wide Area Networks (LPWAN) are a key enabler to enhance the mobility of connected objects [1]. Conversely mobility will significantly increase the number of usages and therefore the number of connected objects. However the Network Operator (NO) coverage restricts devices' mobility. Thus exploiting LPWAN's full potential requires to consider roaming scenarios at least between NOs deploying the same technology.

Roaming refers to the service provided by an NO making possible for a device to securely use a Visited Network (VN), which may be in another country (international roaming) or in the same country (national roaming) when it is outside its regular area. While roaming is an essential pillar of cellular and Wi-Fi networks, it is still in its infancy for LPWAN using unlicensed frequency bands.

Of all LPWAN technologies, the most popular are: (i) SigFox, a single network where devices are identified globally, allowing objects to move from one country to another without any limitation but the regional band used. (ii) 3GPP (4G and 5G) that embeds some roaming capabilities allowing providers to accept devices from other NO in their network. Finally the LoRaWAN technology suffers from a strong network "balkanization". The cheap gateways and the openness of the protocol make it easy to deploy a network in a specific area. Larger networks are often limited to a country, and global initiatives such as The Things Network (TTN) or Helium do not offer good coverage for many applications.

LoRaWAN [2] follows a star topology. Radio Gateways (RG) catch messages broadcasted over the air by devices which relay them to a central point, the LoRa Network Server (NS). It manages devices, authenticates them, and forwards their data to Application Servers (AS), where they are processed. To enhance coverage LoRa Alliance standardised a roaming procedure allowing devices to join a VN.

In this paper, we propose an extension of the current roaming architecture proposed by the LoRa Alliance. Our approach relies on the introduction of new entities called ***Brokers*** managing the relationship between NO and device owner (DO). The concept of Broker improves the scalability and the security of the roaming architecture; it removes the need for a DO to manage complex identifiers and also allows to reduce the number of exchanges to locate a serving NS. As in the current architecture DNS is a key element.

The rest of this paper is structured as follows. Section II provides an overview of the different existing roaming architectures from cellular networks to Wi-Fi and IoT. Section III is dedicated to the different activation modes of a roaming device. Section IV describes our proposed architecture and its implementation. Section V analyses the performance of our solution and compares it with IoTRoam. Finally section VI concludes the paper.

II. Roaming architectures

Roaming allows any NO extending its coverage area. It is indeed a key figure leading to a an extensive adoption of the 3GPP standards and can be taken as example. Inter-NO roaming problem necessitates the definition of roaming agreements consisting essentially of a regulatory component, an economic component, and a technical component [3]. In what follows, we recall this last point in the context of cellular, Wi-Fi and LPWANs focusing in this last case on LoRaWAN.

In cellular networks, mobility management of terminals is based on permanent location updates relative to radio cells in the NO's access network [4,5]. This information is stored in a central database and in another database located at the edge of the network adjacent to the cell area where the terminal is currently attached. These databases enable real-time access to the location information of a terminal to route an incoming communication service to it. The location update procedure supports the mobility of a user who can thus move seamlessly from one cell to another and incidentally from one access area to another, possibly during the execution of a communications service usually defined as a hand-over. The intra-NO roaming can be extended to inter-NO roaming case modulo the roaming agreements including the three components mentioned above, established beforehand between NOs.

Wi-Fi uses WPA2 being a robust authentication procedure based on 802.1X architecture [6]. Once authenticated in the WLAN, a terminal roams from one access point to another without supplementary procedure. Eduroam is an inter-domain Wi-Fi roaming solution [7]. It has been successfully deployed in Europe amongst educational institutions organised as a federation reaching more than twenty thousand locations [8]. It rests on a hierarchical tree of Radius servers the leaves of which are defined by the local Radius servers of each domain implied in the federation. When a terminal wants to access a visited domain, it sends a request to its local access point which will forward it through the Radius tree until its home domain which will authenticate the roaming user and authorise the access of the roaming terminal.

In [9], the authors propose a new mobility management method for IoT. It aims at ensuring and maintaining the continuity of exchanges after changing the link-layer technology between devices and the AS. By using this method, devices may switch from LoRaWAN to NB-IoT. Despite its originality, the approach seems offering limited scalability as it holds on a centralised architecture.

In [10], the authors proposed a roaming mechanism where 5G is used for authentication and key management of devices. Mobility and roaming capabilities of LoRaWAN are in this way extended to a global scale. The results of the experiments performed on a test-bed proved its feasibility. Nevertheless, the device needs to implement two radio technologies increasing its production costs together with its energy consumption.

To handle communications in maritime environments while addressing congestion issues, the authors in [11] propose a roaming technique using LoRaWAN. A distribution server (DS) is introduced to manage roaming of transport vehicles and the follow-up of shipments between two companies. An AI module is feeded with collected information in order to make the right decisions about the traffic. The roaming solution used is not described, security issues are not tackled and the results of their experimentation are not provided.

In [12] the authors have extended the LoRa Alliance's architecture [13] by introducing two components: (i) a Master Agent (MA) managing the communication between two NOs. It provides the `NetID`, which is used to identify a NO while roaming, and (ii) a Local Agent (LA) per NO (vLA and hNA respectively). The MA uses the DNS to identify the different NS from their `NetID`. Then, in order for a device to use the VN, this one must first be provisioned with a `JoinEUI`. The MA uses `JoinEUI` to find the hNS. Although the solution scales thanks to the DNS protocol, it rests on the `JoinEUI` which, as we will see later, induces several issues. The authors focused only on handover roaming.

The LoRa Alliance has defined the most popular and deployed approach for roaming [13] which is valid for version 1.04 and 1.1. devices. The next section recalls it in detail.

III. LoRaWAN Evolution in the Context of Roaming

LoRaWAN is an asymmetric network, where most of the traffic originates from the devices. These ones typically transmit at a very reduced data rate. The network follows a star-like topology. Data transmitted from the device is first received by an RG, whose only function is to forward LoRa packets to the Network Server (NS). RGs have restrictions on packet processing and a significant restriction on duty-cycle. Thanks to this, the cost of implementing a LoRaWAN cell is very low compared to other cellular networks. Until recently, roaming in LoRaWAN was based on independent developments for specific implementations. To enhance network coverage, the LoRa Alliance standardised a roaming procedure allowing a device to use a VN [13].

Fig. 1 depicts the general roaming architecture as in the specification [13]. An NS can play three different roles depending on the type of roaming involved. Namely: serving NS (sNS), home NS (hNS), and forwarding NS (fNS). The sNS controls the MAC layer of the device, the hNS is where the device profile is stored; it has a direct link with the Join Server (JS) that will be used for the Join Procedure and it is connected to the AS. fNS is the NS managing the RGs. There may be one or more fNS. When hNS and fNS are separated, they are in a roaming agreement. Uplink and downlink packets are forwarded between the sNS and the hNS.

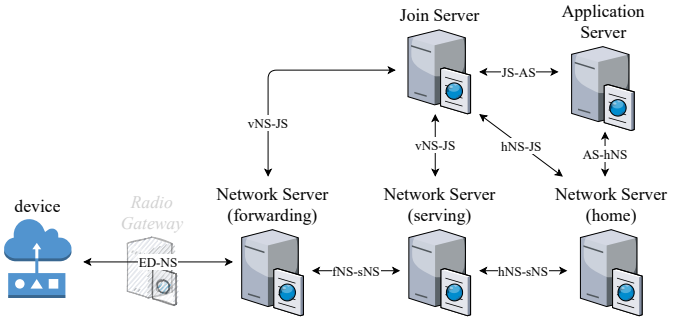

Fig. 1: LoRaWAN Architecture when device is in roaming

The specification describes roaming procedures for ongoing LoRa sessions and for device activation. There are two kinds of roaming: *(i)* ***passive roaming*** where the LoRaWAN session and the MAC-layer control of the device are maintained by the

hNS (*the hNS acts as sNS*). Frame forwarding to and from air interface is managed by fNS; and *(ii)* ***handover roaming*** where the MAC-layer control of the device is transferred from one NS to another (*the fNS acquires the role of sNS*). Therefore, the visited NS (vNS) is either called sNS in the case of handover roaming or fNS for passive roaming. In this article we improve this passive roaming for device activation by introducing a broker to manage authentication of NS.

A. Activation Procedure

In LoRaWAN, the activation procedure aims at delivering a `DevAddr` to each device and at creating session keys from a shared key using AES. There are two types of device activation: Activation by Personalisation (ABP) and Over the Air Activation (OTAA). In ABP, all the necessary parameters have previously been inserted in the device, while in OTAA they are created dynamically. For practical and security reasons, OTAA is preferred over ABP.

To perform an OTAA activation, each device is provisioned with three or four parameters: two IEEE EUI-64 IDs: a device ID (`DevEUI`) and a JS ID (`AppEUI/JoinEUI`), and depending on the device version one or two AES-128 keys: `AppKey` and/or `NwkKey`. At the end of the OTAA, the following information is stored in the device:

- `DevAddr`: a 32-bit address used to identify the device. Its value consists of `NwkAddr` (device address within a network) prefixed by a `NetID`, which is a 24-bit value provided by the LoRa Alliance to identify the NO.
- LoRaWAN 1.0: two sessions keys derived from the `AppKey`: `AppSKey` for encryption and `NetSKey` for integrity.
- LoRaWAN 1.1: in addition to the `AppSKey`, three sessions keys are derived from the `NwkKey`: `SNwkSIntKey` and `FNwkSIntKey` for integrity and `NwkSEncKey` for encryption.

In this article, we focus only on OTAA. For passive roaming, as shown in Fig. 2, this procedure works as follows:

a) The device sends a `Join-request` (JR). Then, RG forwards the JR to the fNS [steps 1–2].
b) The fNS determines whether it is configured to work with the JS identified by the `JoinEUI` contained in the `Join-request`, by doing a DNS query to get the JS IP Address [steps 3–4].
c) The fNS determines if there is a roaming agreement with the respective hNS/sNS by sending a `HomeNSReq` message to the JS. If it exists, the JS replies with a `HomeNSAns` with the `NetID` [steps 5–6].
d) The fNS uses the DNS to lookup the IP address of the hNS/sNS based on the `NetID`. This step can be avoided if the information is stored in the DNS cache. [steps 7–8].
e) The fNS sends a `ProfileReq` message to the hNS/sNS carrying `DevEUI` [Step 9].
f) The hNS/sNS sends a `ProfileAns` message, indicating the type of roaming profile. In this case, it corresponds to Passive Roaming [step 10].
g) The fNS, sends a `PRStartReq` message carrying the `Join-request` message. Then, hNS/sNS forwards the `JoinReq` message to the JS [steps 11–12].
h) The JS replies with a `JoinAns` message carrying the `Join-accept` [step 13].
i) The sNS/hNS sends a `PRStartAns` message to the fNS carrying the `Join-accept` [step 14].
j) The fNS finally sends a `Join-accept` to the device [steps 15–16].

As detailed later in section V, it should be noted that an arrow may represent several exchanges, either in DNS queries to locate the effective DNS server, or in HTTPS communications to open the TCP connection and TLS keys.

B. DNS Resolutions

As described in Section III-A, the DNS service is called twice. One DNS query finds the JS associated with the `JoinEUI` (Fig. 2 steps 3-4), and the other, the sNS from the `NetID` returned by the JS (steps 7-8). So two DNS Zones: **`NETIDS.lorawan.net`** and **`JOINEUIS.lorawan.net`** are needed. The DNS suffixes are obtained as follows:

- The `JoinEUI` represented in the hexadecimal format is first reversed. Then, periods are inserted between each nibble and the domain name.
- The `NetID` (24-bit Unique Identifier) of the sNS.

Recalling that LoRa Alliance is the organisation managing the DNS authoritative servers and maintaining the zone files for both resolutions. As a result, any NO willing to have roaming needs to obtain a **`NetID`** from the LoRa Alliance, and each roaming device must be provisioned with: **`DevEUI`**, **`JoinEUI`** and **`AppKey/NwkKey`**.

The current LoRaWAN architecture is based on public certification to authenticate JS and fNS during TLS exchanges (steps 5-6 and 9-10 of Fig. 2). In IoTRoam [3] initial experimental set-up, it was proposed that the LoRa Alliance acts as the Root CA and generates Intermediate CA to each network. The Intermediate CAs will, in turn, generate leaf certificates for individual servers (JS, NS, AS).

C. Applicability issues

If the roaming procedure described by the LoRa Alliance is working and allows roaming of objects efficiently, some scalability issues remain. If the `DevEUI` allocation is done by the device manufacturer, the DO is responsible of the `JoinEUI` allocation. The `DevEUI` is static, allocated during the device manufacturing, and remains the same during the device lifetime. The `JoinEUI` has to be changed when a device is bought or sold to another customer.

- DO needs to obtain a valid and unique `JoinEUI`. As defined by the specification, these identifiers are allocated by IEEE based on EUI-64, as the first 24, 28 or 36 bits. The remaining bits create globally unique values. It is not compatible with the number of `JoinEUI` a company will need. Despite the costs, the allocation process is defined by IEEE targets object manufacturers, not DOs. LoRa

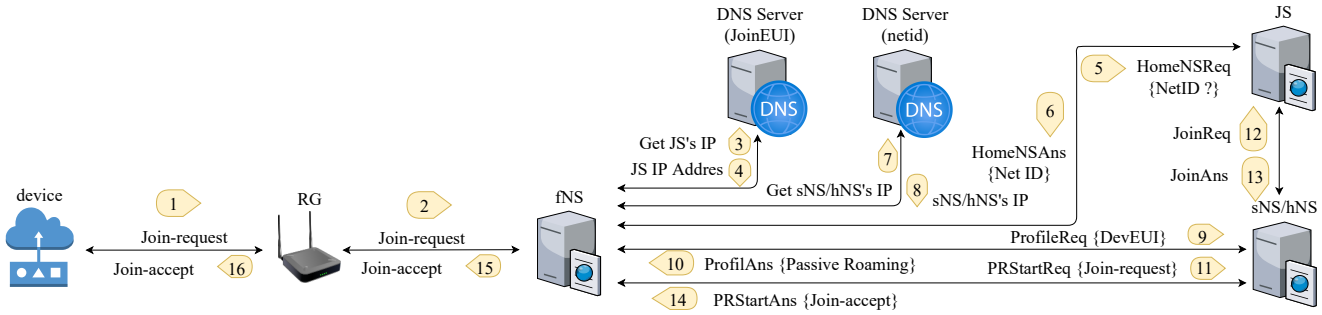

Fig. 2: Join Procedure for Passive Roaming in LoRaWAN.

Alliance may manage `JoinEUI` allocations, yet it creates overhead administrative costs.

- DO needs to insert this `JoinEUI` value in its device. This implies installing new firmware on the device or some provisioning procedure. This may increase the device's costs with more complex software.
- The DNS zone is handled by the LoRa Alliance to ensure the mapping between `JoinEUI` and JS or `NetID` and NS. LoRa Alliance must control who is updating the DNS record. This implies some authentication procedure.
- If the JS is internal to the NO, only the NS will access it to authenticate devices. With the roaming procedure, fNS queries the JS to obtain the NO `NetID` (*[step 5-6] of Fig. 2)*. This may create some security issues since the JS may not be able to forecast which fNS will send a query and expose a sensible element to the internet.

For DO with large fleets of objects likely to perform regular passive roaming, this approach while remaining insecure, costly and difficult to implement, may seriously limit the economic benefits that could be derived from services exploiting the advantages of mobility of connected objects.

IV. Proposed Roaming Architecture

The main technical point regarding passive roaming is to allow any fNS to determine the sNS of any given roaming device as simply as possible. We propose for that an extension to the IoTRoam architecture by introducing a new entity called ***DNS Broker*** whose role is to securely provide this information to entities requesting it. Communication with the Broker is protected through the use of certificates individually allocated by the Broker to DO and NO. They can be limited in time and could be the basis of a commercial service.

The Broker concept introduces some kind of private and secure DNS resolution capability reserved mainly for DOs as economic actors involved in IoT services based on objects' mobility. DOs concerned by these issues can simply feed this information to the Broker. They have to register the `DevEUI` of their objects in a private DNS zone and synchronise it with the Broker using zone transfer capability. The Broker uses this one to create its own zone that will point out to the DO's NS. The Broker will so have a full list of `DevEUI`, this may lead to billions of entries but DNS with zone such as `.com` are able to manage such volume of entries.

The principle of passive roaming can be simplified as follows: when a new device tries to join through passive roaming a NO, the concerned fNS has just to query the Broker to determine the right sNS. The detailed join procedure is presented in Fig. 3, and works as follows:

i) The device sends a `Join-request` (JR) message. RG forwards the JR to its NS (the fNS) [steps 1–2].
ii) The fNS determines whether there is a roaming agreement with the hNs' NO by doing a DNS query carrying the `DevEUI` to get the hNS/sNS `NetID` and IP Address (see section IV-A for further details) [steps $3' - 4'$ and $7' - 8'$]. Steps 5-6 from Fig. 2 which are no longer necessary, are removed.
iii) The fNS sends a `ProfileReq` message to the hNS/sNS carrying `DevEUI` [Step 9].
iv) The hNS/sNS sends a `ProfileAns` message, indicating the type of roaming profile. In this case, it corresponds to Passive Roaming [step 10].
v) The fNS, sends a `PRStartReq` message carrying the `Join-request` message. Then, hNS/sNS forwards the `JoinReq` message to its JS [steps 11–12].
vi) The JS replies with a `JoinAns` message carrying the `Join-accept` [step 13].
vii) The hNS/sNS sends a `PRStartAns` message to the fNS carrying the `Join-accept`. Then, fNS sends a `Join-accept` to the device [steps 14–16].

A. DNS Brokers

As shown in Fig. 3, our roaming architecture introduces a new network element called DNS Broker. Therefore, each time a `Join-request` message carrying an unknown `DevEUI` arrives at the fNS. The fNS performs a DNS query containing the `DevEUI` to get the corresponding `NetID` and hNS's IP address. These two values are used later to establish an IP connection between both NS.

This DNS resolution is divided into two parts: a public part where we resolve the IP of a DNS Broker, and a private part where the `DevEUI` is resolved into a `NetID/IP Address`. By doing this, we bypass `JoinEUI` resolutions in order for NO to reduce the cost of network deployments and to simplify the device provisioning phase.

The second part of the resolution is handled by the **DNS Broker**. Its main role is to handle `DevEUI` and `NetID` DNS

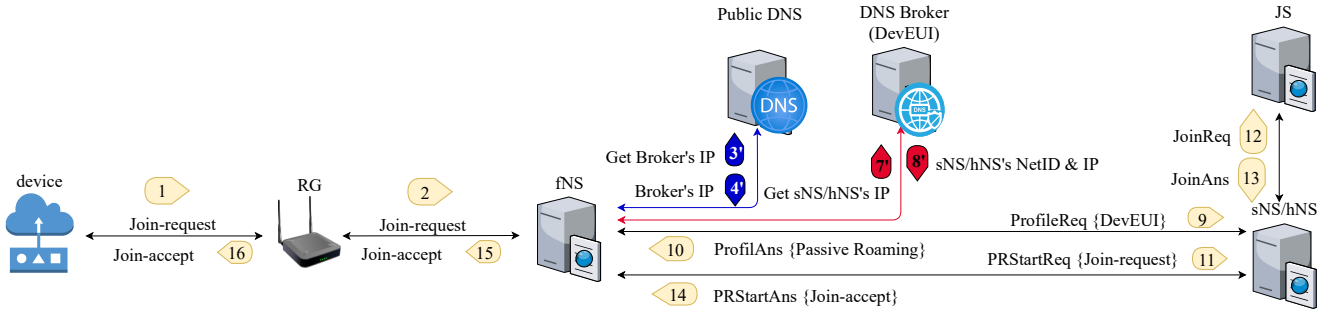

Fig. 3: Roaming Architecture: Join procedure in the proposed Roaming Architecture

Zones, and to deliver client-side certificates. It is then defined as an entity between the NO and the DO. The DO registers its devices at the Broker. The NO queries the Broker when a new device joins the network to allow it or not.

Fig. 4 schematically represents the global architecture with the Broker. DO adds into a database the `DevEUI` of devices that need to be considered. This database is synchronised with the Broker's database. When a new device appears on a NO and sends a `Join-request` message, instead of using the `JoinEUI` to locate the sNS. The fNS contacts the Broker to get the sNS address and continues the joining protocol.

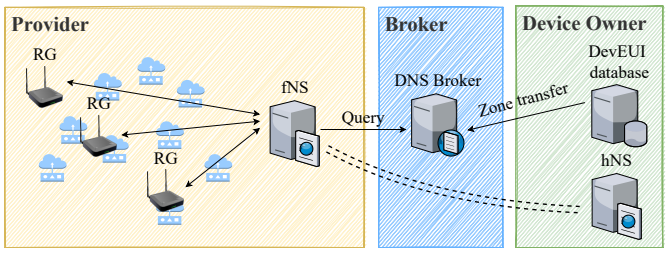

Fig. 4: DNS Broker architecture

The LoRaWAN protocols are not strongly modified, only the fNS implementation has to be slightly modified to use the `DevEUI` identifier instead of the `JoinEUI`. Contrary to the original roaming approach, which implied to centralise some information in DNS zone, namely `NetID` and `JoinEUI` registry, this approach is more flexible.

The needs of our PoC lead us to design a single centralised Broker. However, the general functional requirements of the Broker and their impacts on its architecture deserve an in-depth treatment. We can indeed easily imagine relevant use cases involving a multiplicity of Brokers categories obeying different organisations (hierarchical structures like DNS, federations like EduRoam, decentralised peer-to-peer approach or independent islands). Conversely a DO may register its devices on several Brokers and a NO may simultaneously send its query to several Brokers.

B. Private DNS Resolutions

Current DNS is public. Anyone at any place on the Internet can ask for the name resolution and obtain a result. With the introduction of new mechanisms such as DoH [14] HTTP encapsulates DNS requests through secured TLS tunnels. The goal is to protect privacy while providing confidentiality and integrity; DNS traffic cannot be specifically filtered since it uses HTTPS and DNS queries are encrypted.

DoH is becoming very popular since all main web browsers include by default DoH for name resolution and big companies such as Cloudfare and google are offering DoH services. However, this leadership can weaken network neutrality since all DNS traffic is being centralised into few servers. This can be avoided if we decentralise the DNS infrastructure.

Current DoH implementation authenticates the DoH resolver through its public certificate. The requester identity is not checked since the philosophy is to maintain privacy. If the user identity could be checked by the resolver, some part of the DNS naming space access will be restricted to some users or applications owning appropriate certificates.

The principle is the following; the domain name is known by the requester to be split into different elements. For instance, to contact the sNS the domain name is composed of a public and a private part (underlined). Therefore, we have: **`DevEUI{EUI.64}.deveui.iot-roam.net`**, and the resolution is done as follows:

i) Public part: **`deveui.iot-roam.net`** is resolved using the local DNS upstream server, the DNS response corresponds to the Broker's IP.
ii) Private Part: **`DevEUI{EUI.64}`** is resolved using the DNS Broker as upstream server. A client-side certificate is required in order to have a successful DNS response.

Client certificates are delivered by the Broker, which also acts as CA. As for the connection between NS, contrary to IoTRoam, we propose this to be done following NO policies without central authorities issuing certificates.

C. Implementation through PoC

The roaming architecture described above has been tested and implemented as follows[1]:

a) NS: a modified version of Chirpstack with a dnsproxy DoH client supporting client authentication.

b) DNS Broker: a physical server with a public IP and a registered domain: `broker.iot-roam.net`.

[1]The source code, as well as a tutorial on how to implement our roaming platform is available here: https://github.com/MarinoMtz/LoRaWAN-Roaming-Tutorial.

V. Performance Evaluation

To evaluate the performance of our architecture, we have analysed the traffic through the fNS during the entire roaming process for the join procedure. By doing this, we: (i) determine the IP, DNS, and UDP traffic volume in a DNS packet when the fNS wants to obtain the IP address of the JS/hNS, and (ii) we also determine the IP, TCP and TLS and traffic volume in HTTPS exchanges.

The results presented in Table I involve the traffic required in IoTRoam against our proposal for each protocol at following three categories: (i) DNS, corresponding to the traffic required to obtain the IP of the different servers (JS, NS, Broker), (ii) HTTPS, corresponding to the traffic generated by the exchange of messages between the servers, and (iii) the TLS granularity, which is the volume of traffic required for the handshake and the data. We also count the total number of packets required to join a device as well as the number of RTTs (round-trip delay time).

TABLE I: Performance Evaluation

Types	Parameter	IoTRoam JR ($n=1$)		IoTRoam JR ($n>1$)		Proposed architecture JR ($n=1$)		Proposed architecture JR ($n>1$)	
		#	Len. [B]	#	Len. [B]	#	Len. [B]	#	Len. [B]
DNS	IP	8	160	0	0	4	80	0	0
	UDP	8	56	0	0	4	32	0	0
	DNS	8	606	0	0	4	332	0	0
HTTPS	TCP Sig'	21	1140	3	156	27	1452	3	152
	TLS	15	18167	6	2811	24	25788	6	2793
TLS Gran.	IP	15	300	6	120	24	480	6	120
	TCP	15	480	6	192	24	768	6	192
	C. Hello	3	858	0	0	3	870	0	0
	S. Hello	3	8198	0	0	5	10862	0	0
	C. Cipher	3	5775	0	0	3	6166	0	0
	App. Data	6	2556	6	2499	13	6642	6	2481
# of packets		44		9		55		9	
RTTs		21		6		24		6	

'(SYN – SYN/ACK – ACK)

For the performance evaluation, we performed several JRs. The first JR ($n = 1$) indicates the first time a device tries to join the VN. Then, JR ($n > 1$) stands for a JR arrived latter from the same device. This allows us to see how both platforms react when the DNS cash is full.

a) JR(n=1): regarding the DNS traffic, we note that there is a decrease in the number of exchanges. Indeed, it passes from 8 to 4. This is due to removing the second DNS query where the fNS gets the sNS's IP from the `NetID`.

On the contrary, for HTTPS, we note that it has increased due to the introduction of server/client certificates for mutual authentication with the Broker. Indeed, the amount of packets goes from 15 to 24, This is reflected in the total TLS traffic load that goes from 18167 `B` to 25788 `B`.

b) JR ($n > 1$): as expected, there is no significant change from one architecture to another. The DNS exchanges remain identical for both architectures (0). As for HTTPS, the TLS load is almost identical and the amount of TLS packets is equal to 6.

VI. Conclusion and future work

We have defined a flexible, scalable and secure architecture for roaming in LoRaWAN networks requiring minor protocol modifications of LoRaWAN or DNS. The process of locating the sNS can be simplified by using in addition of DNS, a new entity called DNS Broker allowing private resolution resting on DoH. They are the keys of scalability and security brought by our approach. This also confirms the new role that DNS could play to take advantage of its robustness and reliability. The DNS Broker deserves however a dedicated future study.

Performance measurements are slightly penalised due to the use of HTTPS. However, other lighter approaches such as DNS over CoAP [15] will be considered as a relevant attempt to lighten the protocol footprint while keeping the same functionality.

An open experiment of this architecture with several partner universities will be opened in mid-2022 thanks to the DiNS consortium.

VII. Acknowledgements

We acknowledge Sandoche Balakrichenan, Antoine Bernard, and their team at AFNIC, the authors of IoTRoam for their fruitful insights while testing their architecture. Andrey Meshkov, CTO at Adguard Team for his valuable feedback for updating dnsproxy to enable DoH with client authentication. This research has been partially funded by ANR project DiNS under contract ANR19-CE25-0009-01.

References

[1] A. Bera. (2021) 80 insightful 'internet of things' statistics (infographic). [Online]. Available: https://safeatlast.co/blog/iot-statistics/
[2] "LoRaWAN 1.1 specification. technical standard," LoRa Alliance, Tech. Rep., 2017.
[3] S. Balakrichenan, A. Bernard, M. Marot, and B. Ampeau, "IoTRoam: design and implementation of a federated IoT roaming infrastructure using LoRaWAN," 2021.
[4] S. K. Siddiqui, *Roaming in Wireless Networks*. McGraw-Hill, 2006.
[5] "5GS roaming guidelines," GSMA, Tech. Rep., 2021, version 4.0. [Online]. Available: https://www.gsma.com/newsroom/wp-content/uploads//NG.113-v4.0.pdf
[6] N. Boudriga, *Security of mobile communications*. CRC Press, 2010.
[7] K. Wierenga, S. Winter, and T. Wolniewicz, "RFC7593: The eduroam architecture for network roaming," Tech. Rep., 2015.
[8] (2021) Eduroam supporting services. [Online]. Available: https://monitor.eduroam.org
[9] W. Ayoub, A. E. Samhat, F. Nouvel, M. Mroue, H. Jradi, and J.-C. Prévotet, "Media independent solution for mobility management in heterogeneous LPWAN technologies," *Computer Networks*, vol. 182, p. 107423, Dec. 2020.
[10] E. M. Torroglosa, J. M. A. Calero, J. B. Bernabe, and A. Skarmeta, "Enabling roaming across heterogeneous IoT wireless networks: LoRaWAN MEETS 5G," *IEEE Access*, vol. 8, pp. 103 164–103 180, 2020.
[11] F. Flammini, A. Gaglione, D. Tokody, and D. Dohrilovic, "LoRa WAN roaming for intelligent shipment tracking," in *2020 IEEE Global Conference on Artificial Intelligence and Internet of Things (GCAIoT)*. IEEE, Dec. 2020.
[12] M. Hamnache, R. Kacimi, and A.-L. Beylot, "Unifying LoRaWAN networks by enabling the roaming capability," in *2021 IEEE 46th Conference on Local Computer Networks (LCN)*. IEEE, Oct. 2021.
[13] L. Alliance, "LoRaWAN 1.1 and Backend Interfaces 1.0 Specification," Tech. Rep., 2021.
[14] P. Hoffman, "RFC8484 : DNS Queries over HTTPS (DoH)," *IETF*, 2018.
[15] M. Lenders, C. Amsüss, C. Gündoğan, T. Schmidt, and M. Wählisch, "DNS queries over CoAP (DoC) - draft-lenders-dns-over-coap-02," *IETF*, oct. 2021.