API-driven Dynamic Roaming Activation in LoRaWAN IoT

Luca D'Agati*, Ilenia Ficili*, Marco Garofalo*, Giuseppe Tricomi*†‡,
Francesco Longo*†, Antonio Puliafito*†, Giovanni Merlino*†
**Università di Messina,* Messina, Italy
{luca.dagati, ilenia.ficili, marco.garofalo}@studenti.unime.it;
{gtricomi, gmerlino, flongo, apuliafito}@unime.it
‡*ICAR-CNR: Istituto di Calcolo e Reti ad Alte Prestazioni (ICAR) del Consiglio Nazionale delle Ricerche Italiano (CNR)*
{giuseppe.tricomi@icar.cnr.it}
†*CINI: National Interuniversity Consortium for Informatics, Rome, Italy*

***Abstract*—The rapid growth of the Internet of Things (IoT) landscape underscores the importance of seamless and efficient device roaming within Long Range Wide Area Network (LoRaWAN) infrastructures. This study introduces a comprehensive approach to enhance LoRaWAN-supported mobility, leveraging three key advancements: the decoupling of Gateways (GWs) from core network infrastructures, the adoption of an adaptive GW-to-network connection protocol, and the enhancement of GWs interoperability. A notable innovation of this study is the development of an API-driven agent that enables packet forwarding via unidentified GWs, eliminating the need for pre-established network agreements. Empirical evaluations showcase significant enhancements in LoRaWAN's roaming capabilities, setting the background for future advancements in IoT network technologies.**

***Index Terms*—LoRaWAN, Roaming, IoT, Gateway bridge, API-driven**

I. Introduction and Related Works

The rapid expansion of the IoT is changing our digital environment, enabling the development of applications that range from smart cities and metering to environmental monitoring and sophisticated agricultural practices. Central to this evolution is the LoRaWAN, which enhances IoT's capabilities through its low-power, long-distance communication features [1]. Despite its potential, LoRaWAN faces significant challenges in facilitating device roaming across heterogeneous network infrastructures. These challenges are particularly critical in projects with constrained budgets, such as grassroots initiatives or incremental urban developments [2]. LoRaWAN versions 1.0 and 1.1 were developed to mitigate these issues, with the latter version improving security functionalities and introducing initial roaming capabilities [3], [4]. However, the broader adoption of LoRaWAN 1.1 has been hindered by delays in technical specifications and certification processes, limiting its widespread adoption [5]. Moreover, uneven device compatibility compels networks to operate within the limitations of the least advanced version supported by their devices [6].

Both academic and industry sectors are actively investigating solutions to address these roaming challenges. One notable advancement is the IoTRoam framework, which employs a federated identifier system to streamline and expand the roaming process [7]. Concurrently, there is ongoing discussion regarding the establishment of a cohesive framework that accommodates different network capabilities without renouncing security or operational efficiency [8].

In response to these complexities, this study introduces an in-depth strategy aimed at improving mobility supported by LoRaWAN. By splitting gateways from the primary network infrastructure, implementing adaptable connection protocols between gateways and networks, and enhancing gateway interoperability. A significant feature of this study is the integration of our proposed roaming solution into the ChirpStack GW Bridge package, as illustrated in Figure 1. This integration not only advances the research outlined in [9] but also augments the GW Bridge's existing capabilities by enabling external interaction through APIs, thus allowing data roaming from unrecognized GW devices to their corresponding Network Server (NS). Our approach capitalizes on API-driven mechanisms to facilitate packet transmission across unregistered gateways, obviating the need for pre-existing network agreements and promoting a dynamic, scalable network ecosystem.

II. Proposed Architecture

This study proposes a three-way strategy to enhance LoRaWAN-supported mobility within the version 1.0 framework. First, we advocate for decoupling LoRaWAN GWs from the core network infrastructure, introducing flexibility in deployment strategies. Second, an adaptive GW-to-Network connection protocol, based on real-time network traffic dynamics, is proposed to extend efficiency in device mobility scenarios. Finally, we introduce a dynamic scheme for facilitating roaming capabilities, removing the need for pre-established agreements, which often limit agile deployment and scalable network growth.

A. API-Driven Gateway Bridge Service

An innovation in this research is the modification of the GW Bridge service, enabling the activation and management of roaming capabilities remotely via API calls. This approach allows dynamic adaptation to changing network conditions and user needs, bypassing constraints associated with device roaming in LoRaWAN networks. By leveraging API calls for precise control and activation of roaming functionalities, this edited service framework improves operational flexibility and scalability of LoRaWAN deployments, enabling a user-centric network ecosystem.

B. *ChirpStack Integration*

The study conducted a thorough analysis of ChirpStack architecture components: the GW Bridge, NS, and Application Server (AS). Our exploration began with examining the system's role in facilitating communications between physical GWs and the NS, highlighting operational challenges faced.

The proposed architecture, illustrated in Figure 1, seamlessly integrates the roaming solution into the ChirpStack GW Bridge package. This integration not only builds on the research presented in [9] but also extends the standard functionalities of the GW Bridge by enabling external interaction through APIs. This enhancement allows data roaming from unrecognized GW devices to their respective NS. Additionally, packet decoding functionalities have been implemented to identify the Device Address (DevAddr) and Network Identifier (NetID) of each device, ensuring precise routing of packets to the appropriate NS following API-driven activation. To further increase system reliability and ensure uninterrupted communication, a local database has been deployed to store the necessary information.

C. *Implementation and Experimental Setup*

To validate our proposed architecture, we implemented the solution using ChirpStack, an open-source LoRaWAN network server stack. The key components of the implementation included a modified GW Bridge, and an API service for managing roaming functionalities.

The experimental setup consisted of two LoRaWAN networks, each with its own NS, AS, and GWs. Devices were configured to communicate with their respective NS under normal conditions. For roaming scenarios, devices could transmit data through foreign GWs, which then used the modified GW Bridge to forward packets to the appropriate home NS.

III. Results, Discussion, and Conclusion

Our experimental evaluation demonstrated improvements in LoRaWAN roaming capabilities, focusing on key performance metrics such as latency and packet delivery success rate. Modifications to the GW Bridge effectively enabled roaming traffic management, and the experiments indicated a minor rise in latency, which remains within acceptable bounds for IoT applications, due to the extra processing involved in roaming. The findings are outlined in Table I.

Moreover, our roaming setup sustained packet delivery rates on par with non-roaming conditions, affirming the efficacy of our approach. The integration of an API for roaming management further enhanced device mobility, network scalability, and operational flexibility.

This study proposed a comprehensive method to increase LoRaWAN-supported mobility by splitting GWs from the core

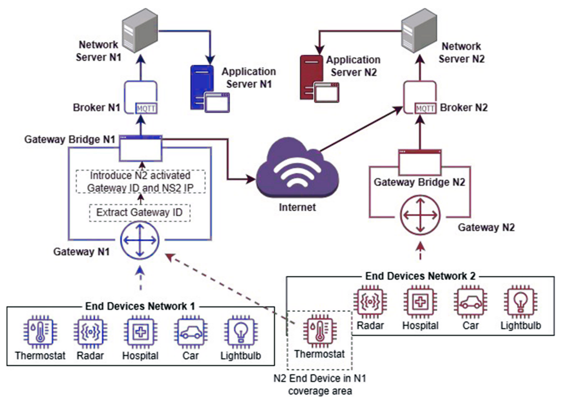

Fig. 1. LoRaWAN server-independent roaming Architecture.

TABLE I
Summary of Experiments

Test Number	**Roaming Devices**	**Home Network Devices**	**Uplink Frequency**	**Avg. Latency at Server**	**Lost Packets at GW**	**Lost Packets at Server**
1	1	2	30 s	359 ms	0	0
2	2	2	30 s	390 ms	0	0
3	3	2	30 s	349 ms	0	2

network infrastructure, implementing an adaptive connection protocol, and enhancing GWs interoperability. Enhancements to the GW Bridge, including API-enabled roaming activation, have extended the network ecosystem. Our empirical analysis underscores these improvements, addressing key LoRaWAN challenges and laying the groundwork for future IoT network advancements.

References

[1] A. Osorio, M. Calle, J. D. Soto, and J. E. Candelo-Becerra, "Routing in lorawan: Overview and challenges," *IEEE Communications Magazine*, vol. 58, no. 6, pp. 72–76, 2020.

[2] I. Butun, N. Pereira, and M. Gidlund, "Security risk analysis of lorawan and future directions," *Future Internet*, vol. 11, no. 1, p. 3, 2018.

[3] "Lorawan® specification v1.0," LoRa Alliance®. [Online]. Available: https://resources.lora-alliance.org/technical-specifications/lorawan-specification-v1-0

[4] "Lorawan® specification v1.1," LoRa Alliance®. [Online]. Available: https://resources.lora-alliance.org/technical-specifications/lorawan-specification-v1-1

[5] "Ts009-1.1.0 lorawan® certification protocol," LoRa Alliance®. [Online]. Available: https://resources.lora-alliance.org/technical-specifications/ts009-1-1-0-certification-protocol

[6] J. R. Cotrim and J. H. Kleinschmidt, "Lorawan mesh networks: A review and classification of multihop communication," *Sensors*, vol. 20, no. 15, p. 4273, 2020.

[7] S. Balakrichenan, A. Bernard, M. Marot, and B. Ampeau, "Iotroam: design and implementation of a federated iot roaming infrastructure using lorawan," Tech. Rep., 2021.

[8] M. M. Sandoche Balakrichenan, Antoine Bernard and B. Ampeau, "Iotroam-design and implementation of an open lorawan roaming architecture," in *2021 IEEE Global Communications Conference (GLOBECOM)*. IEEE, 2021, pp. 01–07.

[9] G. Merlino, R. Asorey-Cacheda, L. D'Agati, F. Longo, A.-J. Garcia-Sanchez, J. Garcia-Haro, and A. Puliafito, "Infrastructure-centric, networkserver-agnostic lorawan roaming," in *2022 IEEE 21st International Symposium on Network Computing and Applications (NCA)*, vol. 21. IEEE, 2022, pp. 149–156.