# Projet Test : Fuzzing du protocole LoRaWAN

Tristan Claverie, Thomas Demany, Manutea Huang

Novembre 2017

## 1 Introduction

LoRaWAN (Long Range Wide-area network) est un réseau radio à haute fréquence (bande UHF, 868MHz). Il est beaucoup utilisé pour la communication des objets connectés et a connu une forte croissance au cours des dernières années (déployé dans plus de 100 pays).

Le fuzzing est une méthode de test consistant à envoyer des données générées aléatoirement à un système et d'analyser son comportement. Il est donc tout à fait adapté au test de en boîte noire d'une implémentation LoRaWAN.

L'objectif du travail présenté est de réaliser un outil de fuzzing se basant sur les cartes de développement LoPy. Ce projet a permis de déterminer que ces équipements possèdent des fonctionnalités adaptées au fuzzing du protocole LoRaWAN. Après avoir présenté le travail réalisé et la démarche suivie, des améliorations sont proposées afin d'améliorer ces équipements au regard de cette capacité.

## 2 Contributions

Le fuzzing du protocole LoRaWAN s'est effectué avec les cartes de développement LoPy. Ces cartes sont basées sur le système sur puce ESP32 couplée avec le transceiver SX7212. Les LoPy implémentent les couches LoRa-MAC (couche 2 du modèle OSI) et LoRaWAN (couche 3 du modèle OSI). De fait, le projet s'est intégralement basé sur ces planches de développement qui ont été à la fois fait tourner les tests (fuzzers) et ont été les équipements sous test (fuzzés).

La réalisation pratique a permis de fuzzer l'implémentation LoRaWAN d'une LoPy en utilisant une autre LoPy qui émettait des trames LoRa-MAC. Ramené au monde IP, cela revient à fuzzer une implémentation d'un protocole de transport (TCP, UDP, ...) en forgeant des trames Ethernet.

Comme tout projet expérimental, les premières étapes ont été consacrées à la compréhension des technologies utilisées et à la mise en place d'un environnement de test. Les différentes étapes de réalisation du projet sont présentées, avec les hypothèses de travail et les résultats obtenus.

### 2.1 Tests préliminaires

La première partie du projet à permis de se familiariser avec l'environnement technique autour des LoPy et de juger de la faisabilité de l'objectif fixé.

#### 2.1.1 Découverte de la LoPy

Comme pour tout équipement inconnu *a priori*, la première étape est de lire la documentation qui lui est associée. Cela a servi à mettre à jour les firmwares des LoPy et de découvrir les APIs accessibles.

La société qui produit les LoPy maintient le plugin Atom Pymakr qui permet de communiquer avec une LoPy. Par défaut, un interpréteur micropython est accessible (via une communication UART) et permet de dialoguer avec une LoPy. Il est aussi possible de transférer et d'exécuter des fichiers entiers via le plugin.

Selon la documentation, il est possible de faire fonctionner une LoPy en mode LoRa-MAC pour envoyer ou recevoir des trames LoRa-MAC ET en mode LoRaWAN pour envoyer ou recevoir des trames LoRaWAN. Ces fonctionnalités sont accessibles depuis l'API micropython des LoPy. La conclusion de cette étape est que le projet est *a priori* réalisable avec les équipements à disposition.

### 2.1.2 Établir une communication en LoRa-MAC

L'objectif de cette étape est de vérifier que le bon fonctionnement des LoPy en les faisant communiquer en LoRa-MAC. Elle permettra aussi de confirmer qu'il est possible d'implémenter un fuzzer sur une LoPy, à savoir de forger et émettre des trames LoRaWAN invalides à partir de la couche inférieure.

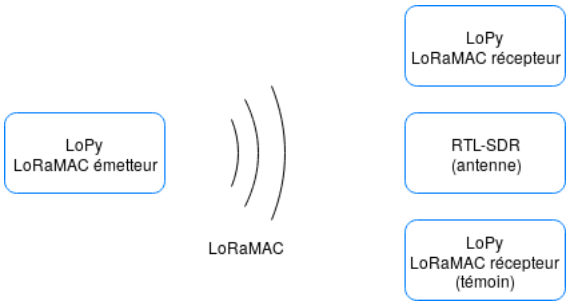


Figure 1: Setup de test

La figure 1 montre le schéma de la manipulation effectuée dans cette étape. Un émetteur envoie des messages 'Ping' à intervalle régulier tandis que le récepteur, sur renvoie des 'Pong' à chaque message reçu. Le récepteur témoin permet d'observer les communications entre les deux équipements, agissant effectivement comme un homme du milieu passif.

Les RTL-SDR sont un type de radio logicielle. Sans rentrer dans le détail, dans ce projet une RTL-SDR a été utilisée à des fins de debug. Elle a permis d'observer le spectre ambiant et de détecter les messages échangés.

La manipulation a permis de vérifier que l'implémentation LoRa des LoPy permet de correctement envoyer et recevoir des messages LoRa-MAC la plupart du temps. Cependant, l'usage de la RTL-SDR a permis de déterminer que la LoPy pouvait tomber dans un état invalide lors de l'interruption brutale d'une boucle (*e.g* Ctrl+C dans un *while True*) quand le transceiver est utilisé. Dans ce cas, aucune erreur ou exception n'est remontée, mais la LoPy devient incapable de réutiliser son interface LoRa. Un *hard reboot* est alors nécessaire.

### 2.1.3 Établir une connexion à LoRaWAN

L'objectif de cette étape est de vérifier qu'une LoPy est capable de se connecter au réseau global LoRaWAN en passant par une autre LoPy qui servirait de gateway. Cela permettra donc aussi de valider que l'implémentation de la gateway utilisée par la LoPy remplit son rôle.

La figure 2 permet de visualiser la manipulation effectuée. La spécification définit plusieurs éléments constitutifs du backbone LoRaWAN. The Things Network (TTN) est le réseau LoRaWAN public donc ce backbone a été choisi dans le cadre des tests effectués.

La manipulation s'est correctement déroulée, à savoir que la LoPy qui simule un équipement communiquant en LoRaWAN a réussi à se connecter à la gateway, qui a correctement transféré les données vers TTN. La LoPy témoin a permis de vérifier que l'émetteur envoyait correctement des données, et la réception des paquets par TTN a été vérifiée par l'interface d'administration fournie.

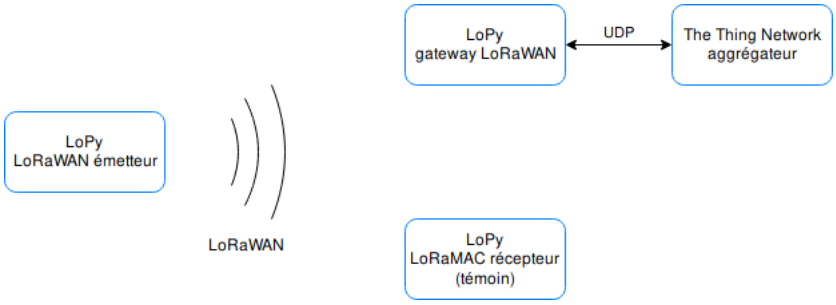


Figure 2: Setup de test

L'architecture d'un backbone LoRaWAN est plus complexe que ce qui avait été initialement supposé : afin de faire fonctionner la manipulation, il a fallu correctement configurer un compte sur TTN. C'est à dire la configuration d'un Network Server (NS), d'un Application Server (AS), d'une application, l'association de la gateway au NS, la création d'une application pour l'AS et l'association de l'équipement spécifique à l'application. Tous ces pré-requis étaient inattendus et la documentation trouvée restait assez laconique sur les manipulations à effectuer.

Lors de la connexion d'un équipement au réseau, la gateway renvoie un datagramme concernant certaines informations de connexion. Cependant, le récepteur témoin n'a jamais réussi à décoder cette trame. Le test a été refait avec un RTL-SDR en plus et la trame apparait bien sur le spectre (la connexion étant réussie à chaque fois, l'équipement qui souhaite se connecter au réseau arrive effectivement à la capter et à la décoder). Ceci semble indiquer que le datagramme est envoyé en utilisant des paramètres de couche physique différents de ceux utilisés par le récepteur témoin. Cependant, aucune tentative de confirmation ou d'information de cette hypothèse n'a été tentée.

### 2.1.4 Synthèse

Les différents éléments des tests finaux ont été validés par ces manipulations préliminaires. Elles ont montré que le fuzzing du protocole LoRaWAN est réalisable à l'aide de cartes LoPy.

Malgré quelques différences entre la documentation et la réalité et différents problèmes rencontrés avec les LoPy et le TTN, cela ne remet pas en cause le projet initial.

## 2.2 Première Version

L'objectif de cette étape est d'observer comment la gateway réagit face à des paquets mal formés envoyés par un client fuzzer à la place d'un client légitime. Cette étape permettrait par la suite de créer des outils d'analyse autour du fuzzing de la gateway.

Le client légitime a été remplacé par un client fuzzer qui envoie des trames mal formées comme schématisées dans la figure 3. Un récepteur LoRaMAC témoin est utilisé pour capturer toutes les trames qui passent, pour s'assurer de l'envoi des datagrammes. La gateway s'assure toujours de renvoyer les paquets à TTN.

D'après la documentation de Pycom, pour envoyer un paquet LoRaWAN, la LoPy doit activer son interface LoRa en mode LoRaWAN. La donnée envoyée est contenue dans une trame LoRaWAN, elle même contenue dans une trame LoRaMAC. De ce fait, pour cette étape, les trames de fuzzing sont des trames de la couche LoRaMAC dont le contenu, payload sur la figure 4, est généré aléatoirement.

### 2.2.1 Synthèse

Les paquets sont correctement générés, envoyés et reçus par l'application. Cependant, il s'avère que la gateway ne réalise aucune analyse du paquet reçu. Elle renvoie le paquet LoRaMAC tel qu'il est reçu à TTN via UDP.

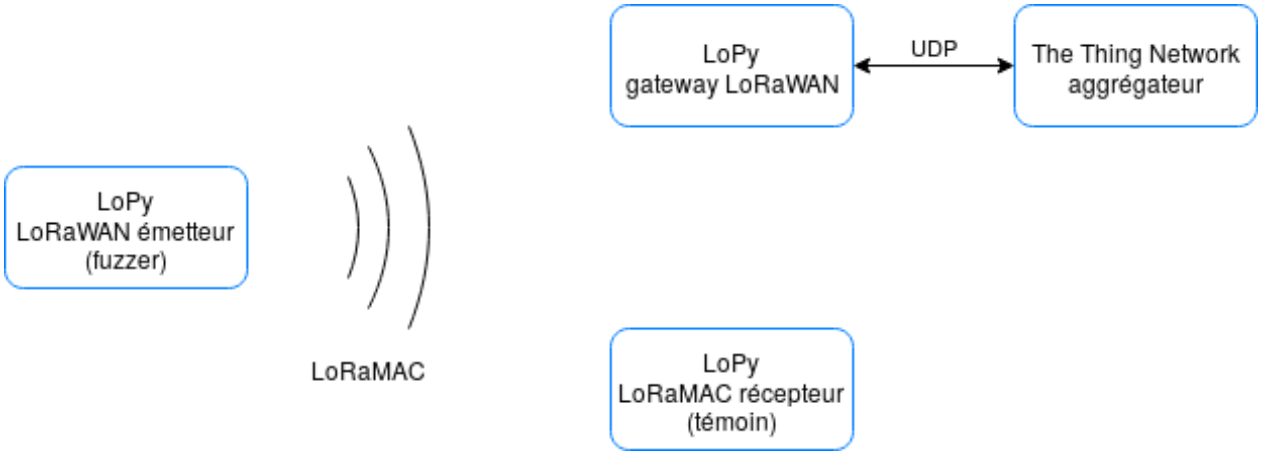


Figure 3: Le client LoRaWAN envoie des trames mal formées à la gateway

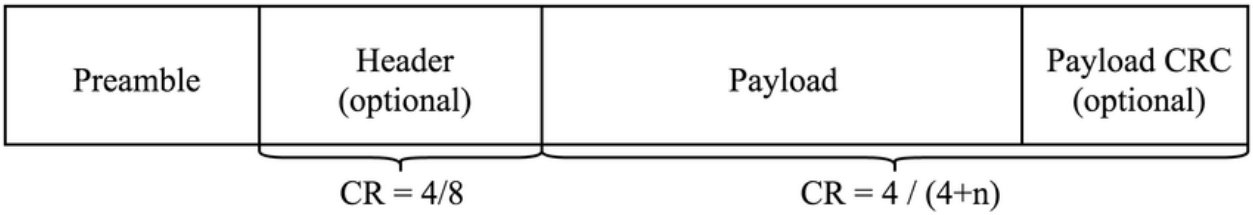


Figure 4: Structure d'un paquet LoRaMAC

#### 2.2.2 Difficultés

La documentation de Pycom associée à la gateway s'intitule "gateway LoRaWAN", cela laissait à penser que la gateway fournie dans la documentation implémentait correctement la norme LoRaWAN. Or, elle ne sert que de proxy entre des équipements et TTN et ne réalise aucune action (selon la documentation, une gateway est supposée avoir un minimum d'intelligence, non présente ici). Par conséquent, l'implémentation de la gateway à disposition n'en est pas réellement une, il n'y a donc aucun intérêt à la fuzzer.

Cela n'invalide pas l'environnement de test, les scripts utilisés et la démarche, seulement l'implémentation fuzzée rend inutile ce type de test.

### 2.3 Deuxième Version

Étant donné que le fuzzing sur la gateway n'est pas possible, il a été nécessaire de modifier le projet.

Pour se connecter à une application, la LoPy envoie une demande de connexion à la gateway. Celle-ci transmet cette demande au broker TTN qui la transmet à l'application. L'application vérifie alors si les clefs de connexion sont identiques aux siennes et renvoie en cas de succès des informations additionnelles (clef de session) pour permettre à la LoPy de se connecter.

La figure 5 présente les différentes étapes et partis mis en jeux dans la demande de connexion d'un équipement LoRaWAN au réseau.

Le fuzzing est toujours effectué sur la couche LoRaWAN, mais cette fois-ci il est prend place sur la réponse envoyée par la gateway à la LoPy cliente. La gateway va alors modifier et remplacer la trame de réponse de connexion par une trame aléatoirement générée.

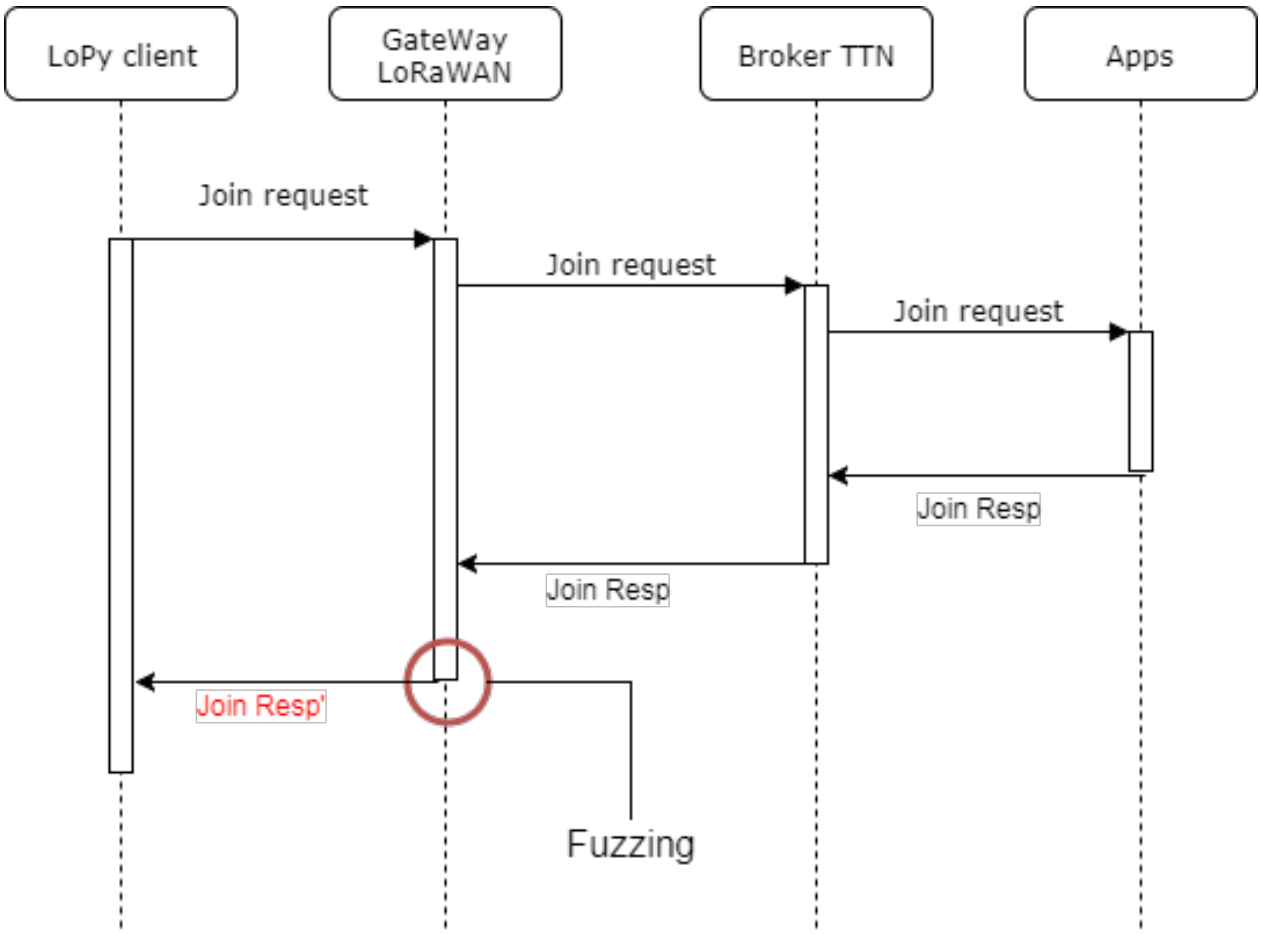


Figure 5: Diagramme de séquence de demande de connexion d'une LoPy client

### 2.3.1 Synthèse

La LoPy reçoit bien les trames modifiées par la gateway mais ne semble pas réagir : Elle les détecte comme étant mal formées et gère correctement les cas d'erreurs.

### 2.3.2 Difficultés

La difficulté rencontrée à cette étape a été le temps d'attente entre les différentes demandes de connexion. Afin de ne pas surcharger la gateway/le TTN les LoPy ne tentent de se connecter que toutes les quelques secondes. Ce comportement est géré par l'implémentation en C de la connexion à un réseau LoRaWAN, ensuite exporté pour micropython.

## 2.4 Pistes d'amélioration

Il y a plusieurs pistes d'améliorations possibles pour tester le protocole LoRaWAN. Une étude plus poussée de la norme LoRaWAN et des types de trames permettrait de fuzzer certains éléments spécifiques d'un datagramme. Une implémentation open-source du backbone LoRaWAN, il peut être judicieux de tester son comportement face à un client/une gateway malicieuse.

Les tests orientés matériels sont aussi intéressants, par exemple pour benchmarker les capacités d'une LoPy vis à vis de certains paramètres (SNR, puissance d'émission, résistance au brouillage, ...). De plus, il peut être intéressant de modifier les paramètres de la couche physique afin de détecter des comportements anormaux chez les récepteurs. Cependant, ce niveau de contrôle n'est pas accessible depuis micropython, il faudrait donc modifier et remplacer le firmware d'une LoPy.

# 3 Conclusion

Le fuzzing est un type de test très puissant pour détecter des problèmes d'implémentations dans des protocoles de communication. Ce projet s'est intéressé au protocole LoRaWAN en particulier. Le fait que LoRaWAN soit un protocole radio complexifie cette démarche puisque les outils de test logiciel standard ne s'adaptent pas à ce cas.

Après avoir explicité la démarche suivie pour ce projet, les différentes étapes de la réalisation ont été présentées. Les résultats des tests effectués sur les LoPy ont été négatifs, à savoir qu'aucun problème d'implémentation n'a été mis en évidence par fuzzing.

Cependant, cela ne remet pas en cause le travail qui a été présenté et les outils qui ont été développés. Des améliorations ont été présentées afin de rendre les LoPy plus efficaces dans une optique de fuzzing. Si elles bénéficiaient de ces modifications alors les LoPy pourraient être transformées en plate-formes d'évaluation de l'implémentation du protocole LoRaWAN dans les équipements clients ou les gateways.