Université Bretagne Sud

Wireless Attack Network Testbed for Embedded Devices

Livrable 1

Projet WANTED

GHARBI Meysoun

Encadrant : Philippe Tanguy

2025/2026

Informations générales

Titre du projet Wireless Attack Network Testbed for Embedded Devices (WANTED) — extension LoRa/LoRaWAN

Type Livrable intermédiaire L1 — contexte, modèle de menace, cahier des charges fonctionnel & technique, rétro-planning

Remarque Livraison L1 (semaine 6-10 et 13-17)

1 Résumé exécutif

Intégrer et orchestrer dans WANTED une suite d'attaques contrôlées pour LoRa/LoRaWAN sur un banc d'essai isolé (testbed). Objectif L1 (fin S2) : chercher et intégrer le maximum d'attaques possibles sur LoRa/LoRaWAN, démontrer des scénarios et fournir logs et documentation minimale.

2 Contexte

Les réseaux LoRa/LoRaWAN sont massivement déployés pour des services IoT. Malgré cela, certaines faiblesses — clés mal choisies, mécanismes anti-replay insuffisants, sensibilité au brouillage radio — exposent des systèmes critiques. WANTED est une plateforme d'orchestration d'expérimentations de sécurité permettant d'exécuter, tracer et analyser des attaques en laboratoire isolé pour identifier des vulnérabilités et proposer des contre-mesures.

3 Modèle de menace (Threat model)

3.1 Portée

Ce modèle de menace ne considère **que des attaques sans fil** (émission/réception RF, manipulation de trames, exploitation protocolaire over-the-air). Les attaques physiques (accès direct au device, sabotage matériel, vol, altération physique des capteurs) sont **exclues**.

3.2 Pourquoi cibler LoRa/LoRaWAN ?

Les réseaux LoRa/LoRaWAN présentent plusieurs attributs attractifs pour un attaquant :

- **Large déploiement** dans des domaines sensibles (compteurs, suivi logistique, agriculture, smart city) offrant de la valeur (données exploitables ou impact réel).
- **Faible capacité des devices** : contraintes CPU/mémoire/power et mises à jour limitées, rendant la correction rapide de failles difficile.
- **Simplicité du lien radio** : modulation LoRa facilement écoutable et injectable via SDR bon marché ; possibilité de brouillage à faible coût.
- **Mauvaises pratiques de déploiement** : clés faibles/partagées, provisionnement OTAA mal configuré, gateways non isolées — permettant compromission par des vecteurs connus.
- **Asymétrie coût/impact** : un attaquant peut causer interruption ou falsification à faible coût matériel pour un impact opérationnel élevé.

3.3 Profils d'attaquant (sans fil)

- **Opportuniste local** : opérateur de SDR grand public (HackRF, LimeSDR) effectuant des expérimentations RF. Motivation : perturbation ou curiosité.
- **Fraudeur distant/local** : cherche à manipuler des données (compteurs, suivi) via over-the-air.
- **Saboteur ciblé** : provoque indisponibilité ou falsification à distance (impact opérationnel).
- **Collecteur / espion** : écoute passive pour exfiltrer des données sensibles transmises en clair.

3.4 Vecteurs d'attaque sans fil (liste priorisée)

Jamming (brouillage RF) émission RF destinée à dégrader la réception sur la bande LoRa (modes : continu, pulsé, sélectif).

Selective / Targeted jamming brouillage temporel ou fréquentiel visant des messages ou périodicités spécifiques (p.ex. fenêtres de join, downlink planifié).

Sniffing / passive eavesdropping capture des trames over-the-air pour analyse (identifiants, métadonnées, payloads non chiffrés).

Replay réémission de trames capturées pour reproduire actions ou créer des états incohérents côté serveur.

Packet injection (over-the-air) création et émission de paquets LoRa/LoRaWAN (join, uplink, downlink) pour influencer le réseau si les protections sont insuffisantes.

Join spoofing / join manipulation manipulation des phases d'OTAA (sous-conditions d'implémentation) visant à tromper le réseau sur l'origine d'un end-device.

MAC-layer manipulation exploitation d'implémentations MAC faibles (ex. counters mal vérifiés, absence d'anti-replay).

Downlink spoofing / Downlink injection — commande malveillante émission de downlinks falsifiés ou injection de commandes over-the-air pour perturber, reconfigurer ou désactiver des appareils (impersonation radio d'une gateway ou injection exploitable).

Duty-cycle / channel abuse saturation de canaux via émissions répétées (conformes/abusives vis-à-vis du duty-cycle) provoquant perte effective de service.

Attaque sur mise à jour OTA (firmware) distribution over-the-air d'un firmware compromis via le processus d'update (attaque du registre d'update, usurpation de source d'update, ou injection de paquets d'update non vérifiés).

Déni de service (DoS / DDoS) sans fil inondation over-the-air (p.ex. flood de join-requests, uplinks massifs) ou épuisement des ressources radio/serveur par volume de messages ou collisions intentionnelles.

3.5 Hypothèses et limites opérationnelles pour L1

- Toutes les attaques sont exécutées en **environnement RF isolé** (atténuateurs/chambre anéchoïque ou VLAN/VM pour la couche réseau).
- Les attaques potentiellement invasives ou à fort impact sont **simulées** (mode `simulate=true`) tant que l'autorisation écrite n'a pas été obtenue.
- Les tests doivent produire des **logs et métriques standardisés** (format JSON) pour permettre une évaluation quantifiable de l'impact.

Remarque **Remarque :** la liste des attaques peut être modifiée selon l'avancement de la recherche et l'accord de l'encadrant.

4 Cahier des charges fonctionnel (extraits prioritaires — L1)

Fonctions principales

F1 – Orchestration WANTED doit pouvoir lancer/arrêter/consulter l'état d'une attaque LoRa via API : `start(params)`, `stop(id)`, `status(id)`.

F2 – Intégration Intégrer les attaques existantes : comme plugins conformes à l'interface WANTED.

F3 – Implémentation Ajouter au moins deux nouvelles attaques avec paramètres configurables.

F4 – Traces & rapport Chaque exécution produit des logs JSON structurés (timestamps, métriques : `packets_captured`, `packets_replayed`, `rssi_avg`, `snr_avg`, résultat).

F5 – Playbook Fournir un playbook de test reproductible (scripts + instructions).

Critères d'acceptation L1

— API d'appel testée pour démarrage/arrêt/status des attaques existantes.
— Démonstration d'un scénario d'attaque sur le banc (logs et mesure d'impact).
— Documentation courte (README, paramètres) et rétro-planning validé.

5 Cahier des charges technique (extraits)

Spécifications techniques

— Interface plugin : `start(params)`, `stop(id)`, `status(id)` — retours JSON.
— Logs structurés (JSON) stockés et exportables.
— Mode **safe** : possibilité de simuler une attaque sans émission RF (tests CI).
— Kill-switch matériel et logique (arrêt immédiat < 2 s).
— Conformité EU868 : respecter duty-cycle, puissance et atténuation en labo.

Matériel et logiciels requis (minimum/ modifiable)

— Gateway LoRa (RAK / dev gateway), 2 end-devices (SX127x ou RN2483),
— PC/VM pour ChirpStack (Network Server local),
— SDR (HackRF) ou radio board programmable,
— Atténuateurs RF ou chambre anéchoïque,
— Outils : Python 3.8+, LMIC toolchain, scapy-radio, GNU Radio.

6 Contraintes & risques

— **Réglementaire** : émission RF strictement contrôlée — respecter la législation.
— **Sécurité & éthique** : autorisation écrite obligatoire ; journalisation complète pour audit.

7 Rétro-planning (1er mois)

S1 (Jours 6–10) — Étude et analyse Étude et analyse du code existant ; intégration initiale et prise en main du dépôt lorawan.

S2 (Jours 13–17) — Recherche intégration ⇒ L1 Recherche des attaques possibles ; intégrer `jamming_simple` et `replay_attack` en plugins WANTED (API + logs) ; exécuter deux scénarios (replay simple, jamming court) ; collecter preuves et rédiger le Livrable L1.

S3 (Jours 15–21) — Implémentation infra isolée Implémentation des nouvelles attaques et mise en place d'une infrastructure isolée (VM + ChirpStack + 1 gateway + 2 nodes) ; installer outils (LMIC, scapy-radio, GNU Radio).

S4 (Jours 27–31) — Validation & documentation Tests de robustesse (timeouts, kill-switch), finalisation du README et des playbooks.

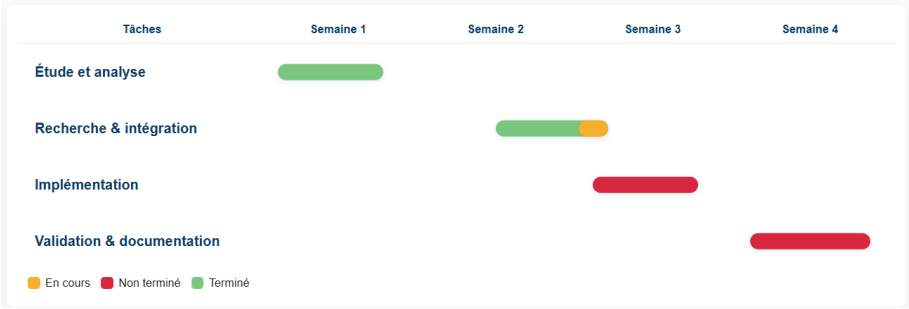

FIGURE 1 – Diagramme de Gantt du projet

Remarque : les durées s'adaptent en fonction des retours de l'encadrant.