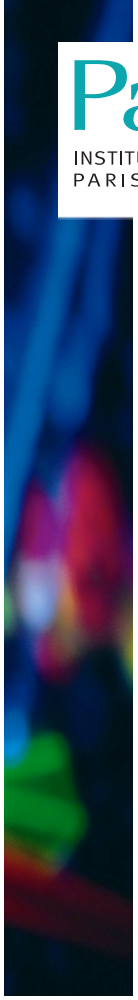


ParisTech
INSTITUT DES SCIENCES ET TECHNOLOGIES
PARIS INSTITUTE OF TECHNOLOGY

TELECOM ParisTech

2013-ENST-00xx

EDITE - ED 130

Doctorat ParisTech

T H È S E

pour obtenir le grade de docteur délivré par

TELECOM ParisTech

Spécialité Informatique et Réseaux

présentée et soutenue publiquement par

Remy LEONE

24 mars 2016

Passerelle intelligente pour réseaux de capteurs

Directeur de thèse : **Dr. Jean Louis ROUGIER**
Co-encadrement de la thèse : **Dr. Vania CONAN**

Jury

Dr. Jean Louis ROUGIER, Professeur, Télécom ParisTech, France	Directeur de thèse
Dr. Vania CONAN, Thales Communications & Security	Directeur de thèse
Pr. Andrzej DUDA, Professeur des universités	Rapporteur
Dr. Nathalie MITTON, INRIA Lille	Rapporteur
Dr. Fabrice THEOLEYRE, Université de Strasbourg	Examinateur
Dr. Thomas WATTEYNE, INRIA Paris	Examinateur
Dr. Marcelo DIAS DE AMORIM, UPMC	Examinateur

TELECOM ParisTech
école de l'Institut Mines-Télécom - membre de ParisTech
46 rue Barrault 75013 Paris - (+33) 1 45 81 77 77 - www.telecom-paristech.fr

T H È S E

A beato torello.

Abstract

Low-Power and Lossy Networks (LLN) are constrained networks typically used in building automation scenarios. To connect natively to the Internet, those networks use a gateway. Because of its key location, at the border of both constrained and conventionnal networks, it can host advanced features. This thesis offers 3 contributions about those problematics and the research methodologies that come with them.

First we offer an adaptative cache that can adapt lifetime of a resource inside a cache in function of the network state and incoming trafic. Proxy cache are used to speed up request treatment. It can be extended by changing lifetime of a resource inside the cache to regulate the amount of request that the LLN have to handle. The optimal lifetime are solution to multi-objective problems. We propose a method based on genetic algorithm to find a set of solution that belong to the Pareto front of the optimal point of configuration.

Second, we propose an estimator of network trafic and energy consumption based on the observation of the network trafic passing through the gateway. Supervision controls and monitors network state. Usually, supervision is done by sending periodic supervision messages. This approache is costly and might be not always possible. By using the network trafic available at the gateway as a base for a model, it's possible to reduce the amount of active supervision needed to monitor a LLN.

Finaly, we propose Makesense, a methodology, coupled to tool ecosystem to create an experiment chain on both simulator and testbed from a unique description. We can combine fast development phase with many iteration then deploying the same code on real nodes to have realistic tests. Once a run is done, the same results analysis toolchain are used. Finally this methodology coupled with continuous integration pave the way to guarantee repeatable experiment.

Résumé

Les Low-Power and Lossy Networks (LLN) sont des réseaux contraints par leurs ressources typiquement utilisés dans des scénarios d'automatisation de bâtiments. Pour se connecter nativement à l'Internet, ces réseaux utilisent une passerelle. Grâce à sa situation à la jointure entre le monde contraint et conventionnel, cette passerelle est à un endroit clé pour accueillir des fonctionnalités réseaux avancées. Cette thèse propose trois contributions autour de ces problématiques de fonctionnalités réseau avancées et des méthodes de recherche qui les accompagnent.

Nous proposons un mécanisme de cache adaptatif permettant d'adapter le temps de vie des ressources dans le cache en fonction du trafic entrant et de l'état des nœuds. Les mécanismes de reverse proxy cache sont utilisés pour accélérer le traitement des requêtes en répondant aux requêtes entrantes à la place des nœuds concernés. Cet aspect peut être étendu en faisant varier les temps de vie des ressources afin de réguler les requêtes touchant le réseau contraint en fonction des ses capacités. Les configurations optimales de temps de validité répondent à des optimisations multi-objectifs. Nous proposons une méthode basée sur des algorithmes génétiques pour trouver le front de Pareto des points optimaux de configuration des temps de validité.

En parallèle, nous proposons un mécanisme d'inférence de supervision du trafic réseau dans le réseau contraint par observation du trafic réseau observé au niveau de la passerelle. Les mécanismes de supervision sont utilisés pour contrôler l'état d'un réseau. L'approche courante consiste à envoyer des requêtes régulièrement afin de connaître l'état d'un nœud. Cette approche est coûteuse à l'échelle de nœuds contraints et doit être limitée. En utilisant le trafic réseau observés à la passerelle comme base d'un modèle, il est possible de réduire le nombre de messages explicites afin de réduire l'impact de la supervision sur le réseau contraint.

Enfin, nous proposons Makesense, une méthodologie couplée à un écosystème d'outils permettant de créer une chaîne d'expériences sur banc de test et simulations à partir d'une description unique. Nous pouvons combiner phase de développement rapide avec des simulations puis déployer le même code sur des nœuds réels afin d'avoir des tests réalistes et une analyse de résultats communes à ces deux phases. Enfin, la méthodologie de développement associée permet d'assurer la répétabilité des expériences.

Remerciements

Je tiens à remercier les jurys, Andrezj Duda et Nathalie Mitton pour les relectures de ma thèse. Je veux remercier également tous les membres de jury et mes directeurs de thèses ainsi que Claude Chaudet, Jeremie Leguay et Paolo Medagliani pour leur aide et leur encadrement durant cette thèse.

Merci à Marc-Oliver et Fabien pour les relectures de mon manuscrit. Merci à Grégoire pour les discussions sur les méthodes d'optimisation multi-objectifs.

Merci à toute l'équipe du LINCS pour les discussions et l'ambiance aussi studieuse que joyeuse Merci à l'équipe de TAI pour m'avoir tant apporté pendant ces trois années.

Je voudrais également remercier chaleureusement Thomas, mon colocataire bien aimé et Marc pour leur soutien à travers les épreuves que ces dernières années m'ont apportées. Je veux remercier ma famille et mes amis qui sont restés à mes côtés pendant ma thèse.

Je voudrais remercier aussi Paris Montagne pour ces années passées à côtoyer le quotidien des chercheurs et la destruction complète de toute autocensure. De même je remercie les communautés liées aux logiciels libres et plus généralement à l'Internet, ma formation n'aurait pas été la mêmes sans les contributions patientes de tous ces gens qui m'ont formé l'esprit.

Table des matières

Table des figures

Liste de tes publis

Chapitre 1

Introduction

We always overestimate the change that will occur in the next two years and underestimate the change that will occur in the next ten.

Bill Gates

Contents

Ce chapitre commence par introduire l'Internet of Things (IoT) (1.1), rappelle les enjeux et domaines d'application (1.2) et les défis apportés par cette nouvelle tendance (1.3). Ce chapitre présente les contributions de cette thèse (1.4) et le plan (1.5).

1.1 IoT

1.1.1 Contexte

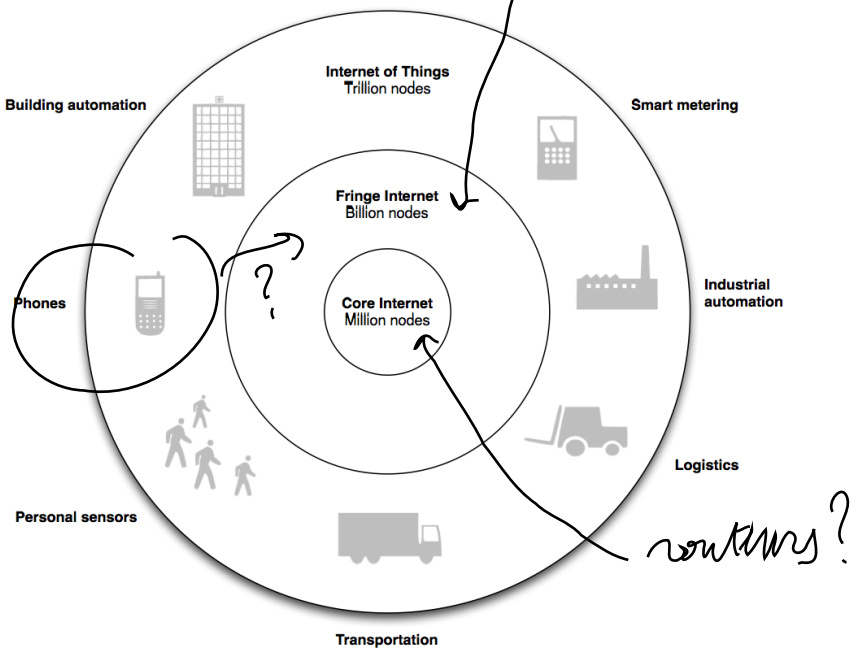


FIGURE 1.1 – Internet des objets (IoT) (Traduire la figure en français)

Internet s'est développé au cours des dernières décennies, partant d'un petit réseau académique pour devenir ubiquitaire et mondial [100]. L'augmentation croissante du nombre d'appareils connectés comme des ordinateurs, tablettes ou smartphones est venu s'ajouter autour des réseaux de cœur pour l'étendre à un réseau de bordure où se trouve de nombreux usages plus mobiles et orientés vers les utilisateurs finaux apportant services et fonctionnalités [39].

La conception de ce réseau avait pour hypothèses de départ les contraintes technologiques de l'époque : des ordinateurs, toujours connectés, alimentés en énergie en permanence et disposant d'assez de capacité pour gérer le trafic réseau reçu.

Ces hypothèses sont toujours valides aujourd'hui pour les serveurs et routeurs composant le cœur de l'Internet et représenté au centre de la figure 1.1.

L'IoT se construit autour des réseaux préexistants montrés sur la figure 1.1 et vient ajouter à des objets existants des fonctionnalités de connectivité à l'Internet. Cette tendance est rendu possible par la réduction constante des besoins énergétiques [63], de la taille, du prix et du poids des processeurs et des capteurs qui a permis d'ajouter une connectivité sans fil à un grand nombre d'appareils. L'objectif n'est pas d'avoir des appareils toujours plus performants mais de permettre une connectivité aussi large que possible pour l'ensemble des appareils permettant de proposer de nouveaux services.

Le nombre de machines et leur versatilité sont en pleine croissance [50] de même que les marchés pour ces appareils aussi bien dans le secteur privé que grand public.

L'essentiel des communications entre ces objets connectés doit se faire sans l'intervention d'humains pour offrir des services très hétéroclites dans des secteurs variés. Les échanges de données entre machines sans l'intervention systématique d'humains sont désignés par le terme Machine to

Machine (M2M).

Ces petites machines se connectent entre elles pour former des réseaux [6] afin d'augmenter la couverture de leur réseau sans augmenter la puissance de leurs communications.

Ces machines sont de nature très diverses, il peut s'agir par exemple d'un capteur, d'une machine industrielle ou de systèmes de contrôle domotiques.

1.1.1 Taxonomie des machines en IoT

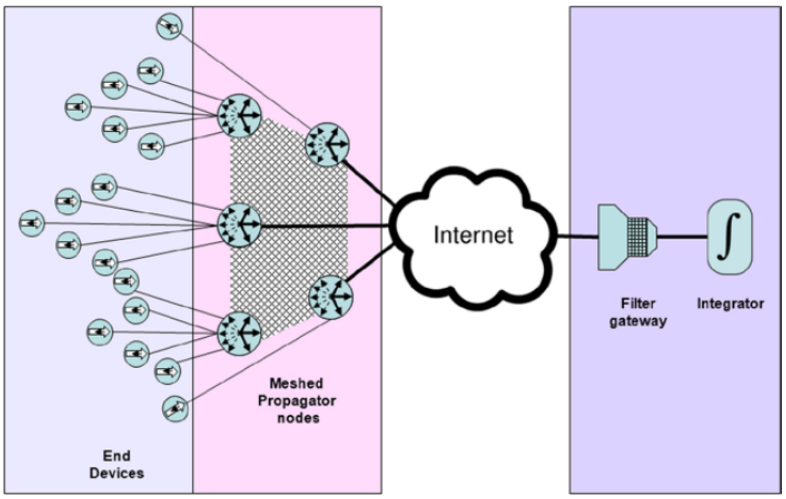


FIGURE 1.2 – Taxonomie des différents nœuds (Figure à refaire pour avoir des légendes cohérentes)

Nous appelons Low-Power and Lossy Networks (LLN)s le réseau formés par des capteurs et une passerelle communiquant entre eux sur des connexions radio bruitées [50]. Certains de ces nœuds ont un rôle de routeur interne relayant les communications des autres nœuds vers la passerelle (aussi appelé routeur de bordure) afin d'augmenter la couverture d'un réseau sans augmenter les portées de transmissions. C'est sur ce type de réseau qui est considéré dans cette thèse.

1.1.2.1 Nœuds capteurs

Afin de saisir l'environnement, il est nécessaire d'avoir des capteurs et actionneurs simples représentés à gauche sur la figure 1.2. Ils sont utilisés dans contextes variés et répondent à des besoins hétéroclites [129]. Ils transmettent et reçoivent des messages courts comme une mesure d'un capteur ou l'ordre de déclenchement d'un actionneur et doivent fonctionner de manière fiable durant des périodes longues.

Les améliorations technologiques sur les composants sont généralement utilisées pour baisser les coûts de production et d'exploitation, notamment en énergie, plutôt que pour améliorer les performances [88], ainsi leurs capacités restent limitées. Les architectures matérielles sont limitées (processeur, mémoire), et la consommation énergétique doit être faible et la batterie doit tenir sur de longues périodes [129]. Pour qu'ils puissent avoir une grande durée de vie avec des batteries simples (piles classiques) les nœuds se mettent en veille autant que possible pour préserver leurs réserves d'énergie.

Afin de masquer l'hétérogénéité des nœuds et des protocoles spécifiques qu'ils utilisent pour se parler, ces nœuds ont besoin de passerelles pour communiquer avec l'extérieur.

1.1.2.2 Passerelles

Ces passerelles (parfois aussi appelée routeur de bordure) sont en charge de concilier un écosystème d'appareils hétérogènes, de collecter les données des nœuds capteurs et d'offrir une interface vers eux. La nature hétérogène des capteurs conduit les passerelles à supporter différents types d'interfaces (par exemple : Courants Porteurs en Ligne (CPL), Bluetooth, cellulaire ou encore Wifi) et protocoles réseau. Elles masquent ainsi la spécificité des capteurs derrière une interface présentant les ressources avec un formalisme commun.

Les passerelles sont plus performantes que les nœuds capteurs et ont un rôle de médiateur comme montré sur la figure 1.2 entre différentes technologies. Du fait de leur position, les passerelles disposent de beaucoup d'informations sur les nœuds capteurs et peuvent en tirer parti entre autre pour établir et optimiser des tables de routage [134] ou découvrir les services offerts par les capteurs [21]. Ces fonctionnalités peuvent s'ajouter à d'autres fonctions (Pare-feux, supervision) et seront détaillées dans le chapitre 2.

1.1.2.3 Infrastructure de services

Bien que les passerelles assurent la connectivité vers les nœuds, l'intégration de ces données est faite au niveau de serveurs distants [5, 6] pour faciliter le passage à l'échelle de sources d'informations multiples. Ces serveurs sont représentés à droite sur la figure 1.2 et représente le dernier chaînon avant l'utilisateur. On y trouve les modules métier à valeur ajoutée en charge de traiter, analyser et visualiser les données dans une interface destinée aux consommateurs et utilisateurs finaux.

Il devient alors possible de construire des services basés sur les données collectées en temps réel comme par exemple des bulletins météo ou des prévisions de trafic routier [53, 50]. Dans cette architecture, ces fonctions d'intégration seront au plus proche des utilisateurs finaux afin que seul les alarmes, exceptions ou notifications pertinentes soient envoyées.

1.2 Enjeux & motivations

1.2.1 Vers de nouvelles fonctionnalités

L'objectif des LLNs consiste à offrir en temps réel des mesures et des relevés sur un grand nombre d'objets avec plus d'aisance de déploiement que ce qu'un système filaire classique peut offrir. Déployés à grande échelle, ces capteurs fournissent des mesures régulières et systématiques de l'environnement permettant des prises de décision plus réactives sur un système complexe. Les LLNs sont utilisés dans des contextes très variés [5, 6]. Les sections suivantes présentent plus en détail les deux principaux champ d'application : la ville intelligente et l'industrie.

1.2.2 Exemple d'applications en milieu industriel

Les systèmes de télémétrie industriels classiques (Supervisory Control and Data Acquisition (SCADA)) fonctionnent le plus souvent en filaire, et dans de grands déploiements cette solution peut être difficile ou coûteuse à mettre en place [5]. Les interconnexions entre les différents systèmes de télémétrie, qu'elles soit physiques (Câble) ou bien logicielles sont très souvent propriétaires, verticalement intégrées et limitées. On assiste désormais à la croissance des communications M2M reposant sur des standards réseau compatibles avec ceux utilisés sur Internet [107].

Les LLN utilisent des protocoles interopérables et des standards ouverts entre tous les équipements de différents constructeurs [125]. Cela permet d'avoir toutes les machines sur le même réseau local ce qui simplifie sa gestion. Au vu du nombre et de la variété d'appareils ainsi que de leurs

constructeurs, l'interopérabilité est un enjeu important et les standards ouverts apportent une solution. Ces déploiements sont sous des contraintes énergétiques strictes et doivent rapporter en temps réel les événements de manière fiable tout en s'acclimatant à des conditions rudes de déploiements. Les canaux de communications doivent être fiables, duplex (on doit pouvoir communiquer avec une machine et elle doit pouvoir répondre sans limites), avec des délais courts et des bandes passantes suffisantes pour des messages compacts. La surveillance d'installations situées le long d'une chaîne de production est un des cas d'applications les plus communs [38, 52].

1.2.2.1 Agriculture et élevage intelligent

Les exploitations agricoles peuvent utiliser les LLNs pour surveiller en temps réel l'état des plantations en terme d'irrigation, la présence de pesticide ou produits chimiques, l'acidité des sols et les conditions météorologiques [128, 104]. La connaissance de l'environnement peut aider à prendre plus rapidement des décisions, de mesurer efficacement au cours du temps l'impact d'un nouveau produit ou encore réduire les interventions manuelles de relevés de mesures dans une exploitation agricole. D'autres systèmes peuvent être utilisés pour surveiller l'état de santé d'un animal, s'assurer que son état et son environnement sont conformes avec les normes en vigueur et s'assurer de sa traçabilité [126] ainsi que sa localisation, ce qui rend les contrôles sanitaires plus rapides, fiables et systématiques sur de grandes échelles.

1.2.2.2 Gestion de bâtiment - Domotique

La surveillance d'un bâtiment [80, 37] est un cas d'utilisation courant pour les LLNs. Le but est d'obtenir au sein d'un bâtiment un système pouvant superviser l'état de l'immeuble sur différents critères comme le chauffage, l'air conditionné, la ventilation et l'allumage des pièces, la fermeture des portes ou la détection de cambriolage[79]. Une fois ce contrôle disponible, il est possible de contrôler le chauffage de manière beaucoup plus automatisée sans l'intervention d'un technicien sur le site et de contrôler plus finement la dépense énergétique pour chauffer un bâtiment.

1.2.3 Exemples d'applications pour les villes intelligentes

Elles sont l'une des applications les plus étudiées des LLN [55, 17], le but de la ville intelligente est de permettre le déploiement d'un système à l'échelle d'une ville ou d'un territoire afin de faciliter la vie de ses habitants. La variété des déploiements et des autorités mises en jeu rend ce contexte différent de celui des applications industrielles où une autorité unique prend la décision d'intégrer un nouveau système à l'existant. Le besoin de standards ouverts est important afin d'éviter des situations d'oligopole sur un système public.

Les systèmes de Smart parking [82] sont un bon exemple de déploiement urbain permettant d'avoir en temps réel la disponibilité des places de parking dans le but de réduire le temps et le carburant utilisé pour trouver une place de stationnement. Ces systèmes sont mis en place en combinant des capteurs détectant des places disponibles sur la chaussée et envoyant ces informations vers des serveurs qui les redistribuent vers des utilisateurs finaux en temps réel. Des services de supervision du trafic automobile et des piétons peuvent également adapter la synchronisation des feux de circulation afin d'améliorer la fluidité du trafic [28]. Dans des scénarios touchant directement les systèmes municipaux (optimisation de l'éclairage public, ramassage des ordures,...), il est d'autant plus facile de justifier la mise en place de ces systèmes, car le retour sur investissement est estimable sur des temps courts [115, 70].

1.2.3.1 Sécurité et Urgences

Des systèmes utilisant des LLNs sont mis en œuvre dans des cas de protection et surveillance de zone dangereuses ou sécurisées [16]. Dans ce genre de situation, les systèmes doivent résister à de nombreuses attaques ou altérations de leur environnement tout en étant capables de répondre en temps réel en cas d'attaque. Ces approches peuvent être couplées avec celle des villes intelligentes dans le cas de surveillances des ponts, des routes ou de grandes structures par leurs vibrations [62]. En outre, la surveillance systématique des infrastructures permet de détecter des problèmes plus rapidement et efficacement. Le déploiement des LLNs permet d'avoir des relevés en temps réel de ces installations. Couplé à des actionneurs, il est possible d'effectuer certaines tâches de maintenance à distance et automatiquement et donc d'éviter de solliciter un technicien.

1.2.3.2 Environnements intelligents

Ce type de système est mis en place dans les cas de préventions et supervisions de risques environnementaux. Les cas d'utilisations classiques sont la détection des feux de forêt [139], les avalanches et glissements de terrain [119, 2], la détection des risques sismiques et volcaniques [129] ou bien la supervision de la qualité de l'eau et de l'air dans des terrains industriels ou contaminés [61, 89]. La surveillance de l'eau [136], d'une rivière ou nappe phréatique influe sur l'approvisionnement en eau d'une ville rendant cette approche d'autant plus pertinente quand l'eau est rare [60]. Ainsi les problèmes survenant sur ces environnement sont localisés plus rapidement, permettant de réagir plus vite en cas d'alerte et d'avoir une cartographie du système en temps réel.

1.3 Défis introduits par les LLNs

1.3.1 Défis liés aux nœuds

1.3.1.1 Hétérogénéité technologique et fonctionnelle

Le grand nombre de constructeurs impliqués dans les LLNs et la multitude des domaines d'applications a conduit à l'émergence de plusieurs standards [8]. Orchestrer un écosystème hétérogène tant en terme de domaines d'applications que de choix technologiques est donc un défi complexe à réaliser.

La tâche est complexe, car différents segments (grand public, industriel), avec différentes chaînes de valeur, différents délais de standardisation (time to market), taille du segment et influence régionale) font que de nombreux standards coexistent et que des outils seront requis pour offrir des interfaces entre ces différents standards. Des organismes nationaux et internationaux se penchent sur les questions d'interopérabilité et de standardisation : Allseen Alliance, Industrial Internet Consortium, ETSI, Open Interconnect, Thread, IPSO Alliance, IEEE...

1.3.1.2 Cycle de vie

Les fonctions d'un nœud capteurs peuvent changer et nécessiter des mises à jour logicielles par exemple pour corriger des failles de sécurité ou ajouter de nouvelles fonctionnalités. Un déploiement rapide des correctifs et des mises à jour est un facteur clé pour éviter l'obsolescence d'un objet connecté et les problèmes de sécurité qui en découlent [92, 127].

Toutefois, il peut être difficile de mettre à jour, par exemple quand le déploiement a lieu dans un terrain difficile d'accès comme un volcan ou une zone de montagne. Des solutions de migrations, de maintenance et de déploiement d'applications à distance sont donc requises [13, 116].

1.3.2 Défis liés au réseau

1.3.2.1 Pertes

Les LLNs utilisent généralement des bandes de fréquences libres (Industrial, Scientific and Médical ISM) pour communiquer afin d'éviter les couts d'une licence pour une fréquence propre. Les canaux étant communs, les liens sont bruités, instables et les pertes de paquets sont donc courantes [7]. Les antennes ont une faible puissance pour économiser de l'énergie ainsi elles sont particulièrement sensibles aux conditions du canal et leur signal peut être écrasé par des antennes d'émission plus puissante comme du Wifi. En outre des phénomènes d'attenuation dus à la réverbération (Multipath fading) se produisent et perturbent les communications mêmes si elles sont à portée de transmission [99].

En cas de trop fortes pertes et de conditions réseau trop instables, les LLNs peuvent adapter la topologie de routage utilisée pour communiquer plus efficacement avec la passerelle [22]. Ceci peut provoquer des reconfigurations dans tout le LLN, ainsi des mécanismes doivent être mis en place dans les protocoles pour garantir la connectivité et la fiabilité des communications entre les capteurs et la passerelle à moindre coût énergétique [1].

1.3.2.2 Connexions intermittentes

Afin d'économiser autant d'énergie et de bande passante que possible, les nœuds se mettent en sommeil régulièrement. Durant cette période leur consommation énergétique est significativement inférieure à la moyenne, le nœud est alors difficilement joignable [59]. Cela entraîne l'intermittence des liaisons radio entre chaque nœud [31] et implique qu'il est particulièrement difficile de savoir si un nœud est en panne, indisponible ou endormi dans le cas où ces cycles de sommeil ne sont pas déterministes. Les appareils doivent donc gérer eux-mêmes les confirmations et la fiabilité des transmissions, car utiliser un protocole de transport classique apporterait trop de surcoûts en entêtes et retransmissions [106]. Ainsi des protocoles sans sessions et des architectures robustes aux déconnexions sont privilégiés afin de transmettre des messages dans le LLN.

1.3.2.3 Besoin de protocoles optimisés

Dans les déploiements classiques des LLNs, les quantités de données échangées sont très réduites et intermittentes afin de préserver l'énergie [117]. Cela requiert d'utiliser des protocoles spécifiques (compact, sans maintien d'état, avec des entêtes compressées) qui réduisent leur impact autant que faire se peut [135, 107, 105].

Les protocoles classiques utilisés dans l'état de l'art pour faire fonctionner l'Internet ont été conçus pour des appareils étant toujours allumés, générant un trafic important et en croissance. En outre, utiliser les protocoles classiques utiliserait une quantité importante de la bande passante et donc un surcoût non négligeable en terme de consommation d'énergie. L'émergence de ces protocoles justifient l'apparition d'une couche applicative d'interopérabilité afin d'orchestrer les différents nœuds sans se focaliser sur leurs spécificités [122].

1.3.2.4 Écoute passive en zone dense

De nombreux protocoles d'accès utilisés dans les LLNs sont asynchrones afin d'éviter de mettre en place une signalisation régulière coûteuse en énergie et en bande passante sur chaque nœud. Lorsqu'un nœud reçoit un signal asynchrone, il doit le décoder et l'analyser afin de décider si cette transmission lui est destinée et si une tâche doit être accomplie. Ce décodage systématique a un coût élevé dans des environnements denses [69].

Ainsi pour des déploiements denses avec un grand nombre de nœuds, il est indispensable d'utiliser des protocoles spécifiques permettant d'avoir aussi bien passage à l'échelle en nombre de nœuds que consommation énergétique minimale lors du décodage des trames émises [1]. Les phénomènes de capture typiques de l'écoute passive sont donc évités et l'énergie n'est utilisé que pour décoder des messages pertinents pour le nœud.

1.3.2.5 Passage à l'échelle

La croissance du nombre d'appareils connecté à Internet reste constante et l'ajout des nœuds venant des LLNs ne fera qu'accentuer cette croissance car baisse continue des coûts de production conduit à des nœuds toujours plus nombreux et moins cher [117, 77]. Cela pose la question du passage à l'échelle de ce type d'installation dans des réseaux qui peuvent déjà être chargés [88, 14, 8].

Afin de répondre à un besoin croissant d'adressage, IPv6 a été introduit afin de fournir un espace d'adressage global suffisant. L'émergence des LLNs rends IPv6 encore plus pertinent car il permet de disposer d'un adressage complet de tous les nœuds sur un réseau commun qu'il soit global ou privé sans utiliser de mécanisme intermédiaire coûteux comme un Network Address Translation (NAT). Cette identification unique permet de facilement fournir une communication sans intermédiaire et simplifie la gestion du réseau par un formalisme commun.

1.4 Aperçu des contributions

La passerelle jour un rôle clé dans la mesure car elle interface le LLN avec le reste du réseau. Elle sert de médiateur et traite donc la problématique de l'interopérabilité des nœuds et des différentes contraintes des capteurs. Sa position dans le réseau lui permet en outre d'avoir une vue précise du LLN et des informations qui y rentrent et en sortent.

Le but de cette thèse est d'exposer comment des fonctionnalités peuvent être ajoutées à l'interface d'un LLN et du reste du réseau pour améliorer l'utilisation des ressources et la connaissance de l'état du LLN. Les contributions proposées dans cette thèse ont pour but d'améliorer les performances et la fiabilité d'un LLN.

1.4.1 Optimisation des ressources du LLN avec un cache intelligent

La passerelle offre une interface vers le LLN et reçoit les requêtes applicatives qui lui sont destinées. Le traitement d'une requête par le LLN est lent car les nœuds sont peu performants, de plus chaque requête consomme de l'énergie qui est en quantité limitée. Ainsi la passerelle a pour objectif de solliciter le LLN aussi peu que possible tout en répondant aux utilisateurs de manière fiable à leur requête.

Une stratégie usuelle pour accélérer les réponses et économiser de l'énergie consiste à maintenir au niveau de la passerelle la réponse à une requête donnée pendant un temps de validité précis. Ainsi si la passerelle dispose d'une réponse encore valide pour une requête donnée, elle l'utilise au lieu de solliciter le LLN. Déterminer le temps de validité d'une réponse pour chaque ressource doit tenir compte de multiples paramètres et la littérature n'offre pas de méthodes explicites pour déterminer des temps de validité admissibles.

Le chapitre 4 propose de déterminer les temps de validité pour réguler le trafic entrant vers les nœuds du LLN en fonction de la durée de vie des nœuds visée par l'administrateur et fraîcheur des réponses attendues. Les objectifs étant antagonistes, il est nécessaire de procéder à un arbitrage afin de trouver un compromis entre différentes solutions. Une modélisation donnée sous la forme d'une optimisation multi-objectives permettra de fournir un ensemble de solutions admissibles optimales.

1.4.2 Mesure implicite de la consommation énergétique d'un LLN

Pour s'assurer que le fonctionnement d'une application est correct, un administrateur a besoin de connaître l'état des nœuds du LLN dont dépend son application. Nous avons vu que pour ce type de réseau, préserver l'énergie consommée était un problème clé. Il parait donc naturel de mettre à disposition des administrateurs un moyen peu consommateur d'énergie leur permettant d'estimer l'état de leur réseau, d'obtenir une durée de vie estimée et connaître les nœuds les plus sollicités dans une topologie multi-sauts.

Mesurer explicitement ces informations n'est pas toujours possible et même quand c'est le cas, il est coûteux de le demander à chaque nœud dans un réseau de grande taille. Ainsi des approches induisant un minimum de transmissions de données peuvent aider à obtenir une cartographie de la consommation énergétique et de l'utilisation de la radio à moindre coût.

Le chapitre 5 montre comment l'observation du trafic routé passant par la passerelle permet d'inférer l'utilisation de la radio, et, dans une certaine mesure, la consommation énergétique. Les limites de cette approche seront également décrites et une correction des biais sera proposée lorsque la supervision active est possible.

1.4.3 Expériences automatisées et reproductibles pour LLNs

Effectuer une expérience avec des LLNs met en jeu de nombreux logiciels et des procédures qui sont longues et fastidieuses lorsqu'elles sont réalisées manuellement. Cela rend une expérience difficile à reprendre surtout si elle n'est pas ré-effectuée par la ou les mêmes personnes. Lorsque de nombreuses étapes sont nécessaires pour obtenir un résultat, documenter et automatiser l'expérience en question autant que possible devient crucial afin qu'elle puisse être reproduite et validée par la communauté scientifique et s'assurer de leur véracité efficacement.

Les testbeds prévus pour les LLNs fournissent de nombreux outils dédiés au lancement d'expériences et à la collecte de résultats [15, 44]. Cependant il n'existe pas de solution complète pour lancer, récupérer et exploiter les données issues d'expériences et de simulations dans un seul outil cohérent et intégré. En outre, l'un des écueils à éviter lors de la conception de tels outils est de définir une architecture trop spécialisée et donc inutilisable dans d'autres contextes ce qui limiterait d'emblée son impact.

Nous proposons un framework d'organisation d'expérience permettant de documenter, d'exécuter et d'analyser l'ensemble d'une expérience sur LLN. Fonctionnant à la fois en simulation et sur nœuds réels, Makesense permet d'obtenir un framework d'expériences reproductibles. Ce chapitre consacré à Makesense illustrera son fonctionnement au travers d'une expérience typique. Il présente entre outre les étapes clés d'une expérience et montre que les choix technologiques ne requiert pas une implémentation spécifique mais utilise au contraire des outils classiques et largement utilisés par la communauté scientifique. Enfin, un mécanisme d'intégration continue automatisera l'expérience et apportera ainsi la preuve de sa reproductibilité.

1.4.4 Collaborations extérieures faites durant la thèse

Un aperçu des collaborations extérieures effectuées au long de cette thèse avec des chercheurs extérieur à notre équipe de recherche est disponible en annexe C.

1.5 Plan du manuscrit

Les contributions de cette thèse exposent différentes fonctionnalités et améliorations que la passerelle peut offrir afin d'améliorer les performances et la fiabilité de ces réseaux.

Le chapitre 2 introduit plus en détail les LLNs. Il présente les hypothèses de travail et les choix faits dans cette thèse pour interfacer un LLN avec d'autres réseaux et les confronte avec ceux couramment trouvés dans la littérature.

Le chapitre 3 présente Makesense le framework d'expérimentation utilisé ultérieurement dans les chapitres 4 et 5 pour documenter, reproduire et partager les expériences effectuée sur les LLNs.

Le chapitre 4 expose comment l'utilisation d'un cache applicatif peut être adapté pour optimiser l'utilisation des ressources d'un LLN en modifiant les temps de validité des réponses des requêtes qu'il reçoit.

Le chapitre 5 montre comment la mesure des temps d'utilisation de la radio dans un LLN permet de prévoir la consommation énergétique implicitement.

Enfin, le chapitre 6 conclue cette thèse et ouvre sur des prolongements possibles.

+ ongleys A, B (&C)

Chapitre 2

Passerelle vers un LLN

We build too many walls and not enough bridges.

Isaac Newton

Contents

Ce chapitre présente et motive les différentes hypothèses et choix techniques au sujet de la passerelle et des protocoles réseaux que nous avons utilisés. La section **??** couvrira les contraintes auquel une passerelle est soumise pour s'interfacer avec un LLN. Puis nous verrons quels sont les contraintes techniques pour les couches basses (2.1.2) celles liées au réseau (2.1) et au routage (2.2.2) et enfin la couche applicative (2.3). Tout au long du parcours de la pile protocolaire nous justifierons nos choix par un état de l'art des alternatives possibles. Nous conclurons ce chapitre (2.4) par un récapitulatif des protocoles choisis et des fonctionnalités que nous proposerons dans les prochains chapitre.

2.1 Interopérabilité réseau

Comme nous l'avons vu dans le chapitre 1, les LLNs ont des applications diverses et s'installent sur des plateformes matérielles très variées. Cependant dans le cas où les nœuds sont contraints, il est nécessaire d'avoir des protocoles spécifiques et un réseau adapté. Ainsi, il existera un équipement réseau à l'interface entre les noeuds contraints et ceux qui ne le sont pas.

Une passerelle est l'unité en charge d'interconnecter deux réseaux de types différents. D'un point de vue matériel, c'est un nœud disposant d'au moins deux interfaces réseaux différentes. Ainsi il peut s'agir d'un système embarqué spécifique, d'un équipement réseau (routeur) ou bien un ordinateur classique faisant tourner une suite de logiciels et disposant de plusieurs cartes réseau spécifiques.

2.1.1 Motivations

Les LLNs seront des "stub networks". Les données sortent et rentrent du réseau par un seul chemin logique et ne font pas transiter des paquets pour d'autres réseaux.

Nous n'étudierons que les approches reposant des réseaux maillés et utilisant du IEEE 802.15.4. D'autres approches de type Low-Power Wide Area Network (LPWAN) sur de longues distances sont développées. Parmi elles on peut citer LoRa et Sigfox. Nous avons souhaité être dans une approche où les nœuds peuvent être interrogés sans limites de requêtes. Les plateformes LPWAN à longue portée sont fortement orientées vers des applications où des informations sont publiées régulièrement et à très faible fréquence.

Ce cas d'utilisation est pris en compte par les protocoles que nous avons choisis (notamment par le `OBSERVE` en Constrained Application Protocol (CoAP)) auquel s'ajoute la possibilité d'interroger un nœud avec une approche proche de celle du web classique tout en offrant des débits plus élevés pertinent dans des cas de surveillance de zone (photo et vidéosurveillance sur des terrains) mais sur des distances de l'ordre de la dizaine de mètres[27]. Enfin, IEEE 802.15.4 permet de s'assurer de l'indépendance vis à vis d'un opérateur particulier.

D'un point de vue fonctionnel, la passerelle est un point de sortie pour un réseau contraint vers un autre réseau. Comme représenté dans la figure 2.3 la passerelle est en charge de faire transiter les paquets et de les traduire dans des protocoles adaptés pour les LLNs qu'elle gère. La passerelle peut être administrée par des interfaces graphiques, web ou textuelle.

Elle est nécessaire et incontournable car les nœuds n'ont pas les moyens de communications nécessaires pour se connecter à des réseaux conventionnels par eux-mêmes en raison de leurs contraintes physiques. Ainsi le rôle de la passerelle est prépondérant pour effectuer une interconnexion efficace.

- Modèles verticaux - Modèles horizontaux

Les nœuds auront des piles protocolaires variées. Afin de se connecter à l'existant ces passerelles devront communiquer sur les technologies usuelles comme 4G, Wifi et Ethernet. Cependant elles devront aussi pouvoir parler des protocoles adaptés aux nœuds contraints comme le Bluetooth low energy [109] ou IEEE 802.15.4. Ainsi il est pertinent de mettre au niveau de la passerelle des fonctionnalités de traductions transparentes de protocoles. Il peut s'agir de protocole réseau comme IPv4/IPv6 qui ont des modes de fonctionnement différents ou bien des fonctionnalités de traductions de protocoles applicatifs tel que HTTP vers des protocoles spécifiques.

2.1.2 Communication vers LLNs

Les solutions de connectivité sans-fils sont très nombreuse dans le champ de l'IoT [112] Lorsque qu'un déploiement filaire est possible, des solutions de Power over Ethernet (PoE) permettent d'alimenter les nœuds et ainsi d'avoir une connexion réseau classique en plus d'une alimentation électrique. Cependant ces cas de figures sont rares car tirer des câbles pour un si grand nombre d'objets

complique leur déploiement et augmente les coûts. Des solutions sans-fils sont donc proposées pour palier à ce problème. Il existe de très nombreux protocoles sans fils, elles offrent différents compromis en terme de portée et de bande passante.

2.1.2.1 Accès longue portée

Initié par Sigfox, les technologies longue portée à bas débit permettent de transmettre à des débits modestes sur des distances de l'ordre de plusieurs kilomètres. L'une des forces de ces technologies consiste à utiliser des bandes de fréquences ISM qui sont libres de droit et permettent pour un coût modique d'avoir de grandes portée avec peu d'antennes. Ainsi Sigfox se place comme opérateur réseau M2M bas-débit. La LoRa alliance utilise également les mêmes bandes de fréquences avec une technologie proche de celle de Sigfox mais n'a pas l'approche de fournisseur d'accès et fournit sa technologie sous licence.

Ces technologies offre un jeu de compromis clair : peu de débit, une grande portée, l'utilisation de bandes de fréquences libres pour réduire les couts d'entrée. Cependant cette méthode d'accès présente des limitations. L'une d'entre elle découlent de l'utilisation des bandes ISM qui stipulent que les objets ne peuvent rester éveillés plus de 1 % du temps [123]. Ainsi si le nombre d'objets devient grand il devient nécessaire de densifier le réseau d'antenne pour répartir la charge ce qui augmente les couts de déploiements. En outre, au vu de ces limitations, le trafic descendant de la passerelle vers les nœuds à un coût très élevé et ne peut être envisagé que dans des déploiement où la capacité des antennes est clairement surdimensionné par rapport au nombre d'objets qu'elles doivent gérer. Ainsi dans le cas où les paquets doivent être acquittés, l'utilisation de ces technologies peut se révéler délicate.

La 5G, annoncée pour les années 2020, entend regrouper et intégrer toutes ces technologies longue portée pour une utilisation plus efficace des bandes de fréquences qui seront disponibles.

2.1.2.2 Accès courte portée

D'autres technologies sans-fils utilisant des bandes de fréquences libres sont également disponibles. Parmi les plus classiques on peut citer le Wifi. Cependant ce n'est pas un choix judicieux dans le cas de nœuds contraints, car il consomme beaucoup d'énergie en plus de nécessiter un composant radio couteux [112]. De plus son débit est très souvent surdimensionnés par rapport aux besoins d'objets contraints. Ainsi des technologies sans fils de faible puissance sont plus adaptés aux scénarios typiques de l'Internet des objets :

Z-wave Z-wave est une spécification complète de la pile protocolaire. Bien implenté depuis de nombreuses années en particulier dans le secteur de la domotique, Z-wave s'est distingué par une retro-compatibilité garantie sur tous les équipements labellisé par la Z-Wave alliance. Cepedant la retro-compatibilité implique une rigidité sur certains points du protocole, notamment le fait de ne pas avoir plus de 4 noeuds relais et ne pas pouvoir avoir des réseaux de plus de 232 noeuds. En outre, en cas de changement de topologie la reconstruction des tables de routage peut être longue [110]. Ainsi Z-wave est recommendé dans des scénarios de domotique de particuliers. Malgré tout, Z-wave dispose de fonctionnalité de réseaux mesh testées que d'autres protocoles comme Bluetooth Low-Energy (BLE) n'ont pas au même niveau de maturité.

BLE Tout comme Z-wave, c'est une spécification complète de l'ensemble de la pile protocolaire. Elle dispose de beaucoup d'atouts : un débit important (1MB/s) permettant de passer peu de temps à émettre, peu d'entêtes, une grande efficacité spectrale et un schéma de saut de fréquence pour éviter les canaux chargés. Cependant même si la construction de réseau mesh avec cette technologie est possible il n'est pas encore standardisé et n'est donc pour le moment pas interopérable [109].

IEEE 802.15.4 IEEE 802.15.4 spécifie à la fois une couche physique et liaison dans le modèle OSI [9]. Elle est conçue pour être utilisée dans des appareils aux ressources limitées pour leur permettre d'avoir une connexion sans fil nécessitant peu de complexité et de ressources. Ce protocole appartient à la famille des protocoles Low Rate Wireless Personal Area Network (LRWPAN). Il permet typiquement des portées de communications de l'ordre de 10 à 50 mètres, permet d'avoir un débit de 250 kbits/s. IEEE 802.15.4 évolue et a intégré récemment des mécanismes de sauts de fréquences pour augmenter la fiabilité de ses transmissions [1].

IEEE 802.15.4 est compatible avec de nombreux autres protocoles réseaux industriels, parmi les plus courants nous pouvons citer Zigbee [3, 113] qui est une spécification complète de l'ensemble de la pile protocolaire, Wireless HART [71] et IPv6 over Low power Wireless Personal Area Networks (6LoWPAN). De plus des nouveaux standards se construisent autour d'elle comme Thread [111] et des travaux sont entrepris pour construire des techniques d'interopérabilité entre elle et BLE [98].

Nous avons choisis d'utiliser IEEE 802.15.4 et d'avoir des réseaux maillés car nous voulons pouvoir solliciter un nœud sans limites de requêtes montantes ou descendantes. Les plateformes LPWAN à longue portée sont fortement orientées vers des applications où des informations sont publiées régulièrement et à très faible fréquence. Nous voulons être dans une configuration où les nœuds contraints peuvent être interrogés de manière non régulière.

Toutes les technologies vues dans ce panorama non exhaustif offrent des compromis différents et à mesure où les couts de fabrications et de déploiement de ces radios baisseront, des passerelles compatibles avec l'ensemble de ces radios apparaitront (ZigBee, IEEE 802.15.4, Z-wave, Thread, Bluetooth Low Energy) de même qu'elle pourront se connecter au réseaux plus classiques via WiFi, Ethernet, LoRa, LTE, 3G/GPRS ...En outre, toute la complexité de ces technologies peuvent être masquées derrière la couche réseau qui a pour but de les interconnecter.

2.1.3 Interconnexion IP & Compression

Le rôle de la couche réseau est de rendre interopérable des liens hétérogènes. A la bordure du LLN, se trouvera une passerelle qui fera office de routeur et qui aura plusieurs connexions avec d'autres réseaux par exemple WiFi, Ethernet ou encore en LTE. En plus de ces fonctionnalités d'interconnexions il est nécessaire d'avoir des fonctionnalités de sécurité comme un pare-feu afin d'empêcher les attaques venues de l'extérieur d'atteindre les nœuds du LLN afin de faire par exemple des attaques au déni de sommeil.

La passerelle devra être accessible depuis des réseaux standards, ainsi elle sera compatible avec IPv4 et IPv6 puisque ce sont les deux protocoles réseaux les plus déployés aujourd'hui. Cependant, les adresses IPv4 s'épuisent et les LLNs comportent un grand nombre de nœuds, ainsi IPv6 pourrait permettre l'adressage de chaque appareil beaucoup plus facilement que IPv4 car il éviteraient les mécanismes de NAT. Ainsi, utiliser IPv6 sur les nœuds simplifierai à la fois la connexion des nœuds à l'Internet mais aussi le processus de développement.

IPv4 est encore massivement déployé et à court et moyen terme IPv6 et IPv4 cohabiteront [23]. Ainsi la passerelle offrira des tunnels entre les hôtes du LLN (utilisant IPv6) et ceux du réseau local qui utiliseraient IPv4. IPv6 apporte une réponse au problème de la pénurie d'adresse en IPv4 par son vaste espace d'adresse disponible rendant NAT superflu. D'autres problèmes d'IPv4 sont également résolus en IPv6 comme l'auto-configuration du réseau supprimant Address Resolution Protocol (ARP) et Dynamic Host Configuration Protocol (DHCP). Cependant comme nous le verrons dans la section 2.1, les paradigmes d'IPv6 doivent être adaptés au LLNs.

- Dire que IPv4 montre ses limites - IPv6 permet un espace d'adressage suffisant - Possiblitité d'avoir des connexions de bout en bout sans tiers comme un NAT

Cependant, IPv6 n'est pas adapté tel quel à IEEE 802.15.4. De plus, le support des entêtes IPv6 sur du IEEE 802.15.4 laisserait peu de place pour le contenu applicatif. Les bandes passantes faibles, les

ressources limitées en énergie et la taille maximale de 127 octets alors que la Maximum transmission unit (MTU) de IPv6 est de 1280 octets sont les contraintes les plus fortes de IEEE 802.15.4.

6LoWPAN [66] rends l'adaptation d'IPv6 au réseau contraint possible sur IEEE 802.15.4. Nous avons voulu par le choix de 6LoWPAN privilégier les technologies et les paradigmes provenant de IP. 6LoWPAN [107] est un sujet vaste, nous ne présenterons ici que les aspects relatifs à la gestion de la topologie et à l'adaptation à la passerelle.

la couche d'adaptation entre IPv6 et IEEE 802.15.4 proposant un format de trame pour la transmission des paquets, une méthode pour définir les adresses en lien local et des configurations d'adresses automatiques, la compressions des entêtes et processus de transfert de trames dans les réseaux maillés [86].

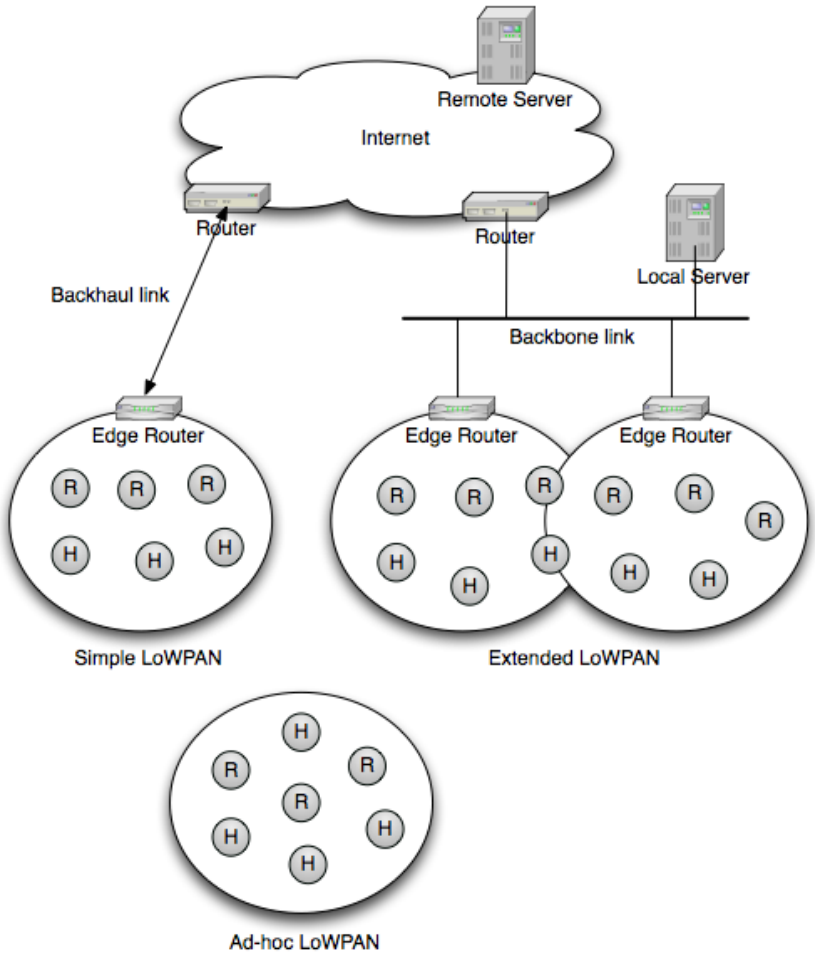


FIGURE 2.1 – Architecture 6LoWPAN

Les LLNs sont présentés comme étant des réseaux "stubby", ils ne font pas transiter du trafic en provenance d'autres réseaux. La Figure 2.1 présente l'architecture mise en avant par 6LoWPAN. Il existe essentiellement trois types de nœuds. Les nœuds (ou 6LNs représentés par un H sur la figure 2.1) dans un Low-Power Wireless Personal Area Network (LoWPAN) utilisent leur capteurs, envoient et reçoivent des paquets mais n'en transfère pas pour le compte d'autres nœuds. Les routeurs (ou 6LRs représentés par un R sur la figure 2.1) sont des nœuds intermédiaires qui font suivre des paquets

entre leur origine et leur destination. Enfin, les routeurs de bordures (ou 6LBRs) sont la connexion entre des nœud dans un réseau 6LoWPAN et d'autres réseaux locaux. Typiquement, les nœuds et routeurs sont contraints en énergie et en ressource de calcul alors que le routeur de bordure ne l'est pas.

Les 6LoWPAN Border Router (6LBR) sont responsables de la dissémination des préfixes IPv6 et de la compression des entêtes à travers le LLN. Ils maintiennent également un cache de toutes les adresses IPv6 et des identifiants EUI-64 afin de pouvoir effectuer des Duplicate Address Detection (DAD).

2.2 Supervision du LLN

2.2.1 Etat de l'art

- WeatherMap - Connaitre la consommation - Connaitre les problèmes dans le déploiement d'un réseau - Savoir quels sont les noeuds disponibles

2.2.1.1 Routeur de bordure

Dans le cas où les nœuds sont connectés à un autre réseau, la passerelle fait office de routeur de bordure et de pare-feu. Elle échange des routes vers les nœuds de son réseaux avec l'extérieur et contrôle les accès aux ressources. Lors du déploiement d'un réseau classique en IPv4 on a une passerelle qui a pour charge d'offrir une connexion à l'Internet. Cependant cette connexion dans les cas usuels de IPv4 repose sur un NAT. Or dans le cas où on doit gérer un très grand nombre d'adresses comme c'est prévu dans l'IoT cette approche n'est pas envisageable. Les NAT ralentissent les traitements et ne sont pas pertinents dans le cas où l'on a une technologie comme IPv6 qui permet un très grand nombre d'adresses. Par conséquent les technologies classiques d'interconnexions entre réseaux doivent être mis à jour pour tenir compte des spécificités de l'IoT. Nous verrons dans la section 2.1 les adaptations spécifiques faites à IPv6 pour s'adapter aux scénarios contraints.

Fonctionnalités de la supervision La supervision désigne la surveillance continue d'un équipement réseau afin de s'assurer que son fonctionnement et ses performances sont celles attendues [74]. Le cas de supervision le plus courant est une supervision active. Elle utilise des messages explicites pour demander l'état d'un équipement. Typiquement il s'agit d'un nœud qui va périodiquement envoyer une requête pour connaître l'état d'un équipement. Une autre approche de la supervision est dite passive quand elle n'utilise que les informations qui sont déjà disponibles sans aucune autre intervention explicite. Typiquement utilisée dans la supervision réseau, il peut s'agir de remontée de traces de trafic réseau qui sont passées par un routeur de bordure ou bien de sondes avancées. Ces sondes peuvent être mises sur la passerelle afin d'avoir une bonne vue sur les flux réseaux entrants et sortants.

Problématique de supervision dans les LLNs Les techniques de supervisions sont aujourd'hui connues et bien étudiées, cependant elles sont basées sur un jeu d'hypothèses assez fortes sur les nœuds. Dans le cas idéal, ils sont toujours actifs et peuvent répondre aux requêtes de l'administrateur. Cependant dans le cas d'un LLN, les nœuds sont souvent endormis et ne peuvent répondre aussi régulièrement aux requêtes en outre ils n'ont pas forcement des mécanismes d'introspection sur leurs états très élevés ou utilisables. Par conséquent les techniques usuelles de supervisions sont coûteuses énergétiquement et pas forcement applicables sur de larges LLNs.

2.2.2 Routage pour LLNs

6LoWPAN n'offre pas de méthode de routage par défaut, or dans des cas de réseaux multi-sauts, un nœud peut être utilisé comme routeur pour faire transiter le trafic de ses voisins. Le but d'un protocole de routage est de construire et de maintenir dynamiquement les routes utilisées par les paquets pour traverser le réseau. Dans le cas d'un LLN, des compromis doivent être faits afin de maintenir un routage efficace tout en évitant de surcharger les nœuds.

La passerelle a un rôle prépondérant pour un protocole de routage puisqu'elle offre un point de sortie pour chaque nœud qui veut envoyer des messages vers l'extérieur. Ainsi les protocoles de routage les plus utilisés sont ceux qui vont privilégier les communications multipoint to point.

Il existe plusieurs protocoles de routage adaptés aux LLNs, parmi lesquels Routing Protocol Layer (RPL) et LoadNg [124]. Nous avons privilégié le choix de RPL [134] qui est un protocole proactif permettant de disposer au niveau de la passerelle de la topologie réseau ; contrairement à LoadNG qui est réactif et qui construit les routes dynamiquement. Nous utiliserons les topologies réseau tout au long des chapitres 4 et 5.

2.2.2.1 Fonctionnement du protocole

RPL est un protocole de routage proactif à vecteur de distance pour LLN. Il construit et maintient une topologie réseau sous forme d'un Directed Acyclic Graph (DAG) ayant comme racine une ou plusieurs passerelles. Les données transmises par les nœuds du réseau ne seront transmises par les liens du DAG. RPL permet le trafic Multi-point to point (MP2P) (appelé trafic montant), Point to Multi-point (P2MP) (appelé trafic descendant) et le trafic Point to Point (P2P).

RPL est disponible et implémenté sur plusieurs systèmes parmi lesquels : Contiki [121], OpenWSN et RIOT.

2.2.2.2 Construction du Destination-Oriented DAG (DODAG)

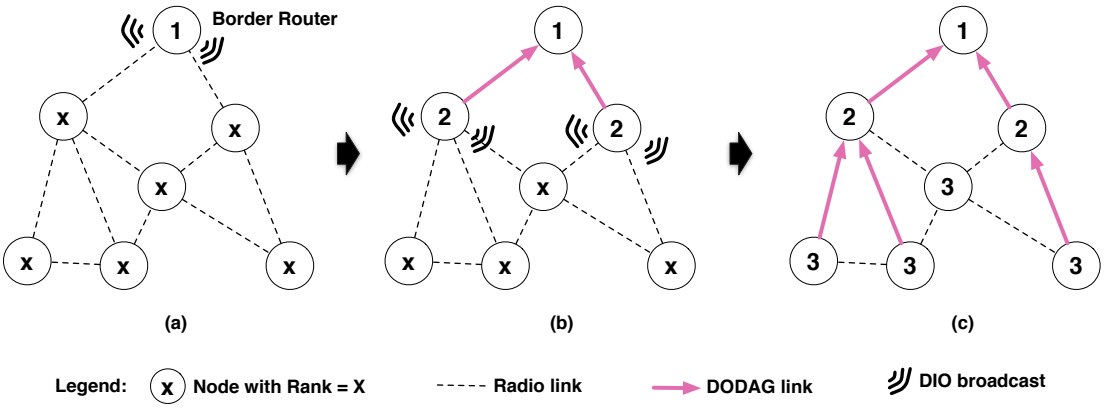


FIGURE 2.2 – Construction d'un DODAG

Le graphe construit par RPL réponds aux besoins décris par les fonctions objectives utilisées. Le processus de construction du DODAG démarre au LoWPAN Border Router (LBR) utilisé comme racine qui est généralement le nœud collecteur de données. RPL supporte l'utilisation de racines multiples configurées dans le réseau. L'ensemble des paquets de signalisation utilisés pour construire cette structure sont des paquets Internet Control Message Protocol v6 (ICMPv6).

Le processus est représenté dans la figure 2.2. La racine commence la diffusion des informations sur le DAG qu'elle veut construire en utilisant un DODAG Information Object (DIO). Les nœuds à portée de communication de la racine reçoivent ce DIO, et rejoignent ou non la structure selon le résultat de la fonction objective, les caractéristiques du DAG et le coût du chemin annoncé). Une fois que le nœud s'est joint à la structure, il a une route vers la racine de la structure DODAG). La racine est appelée le parent du nœud. Le nœud calcule son rang dans le graphe, qui représente la position du nœud dans la structure DODAG. Chaque nœud dans le graphe a un rang qui représente la position relative de ce nœud par rapport à la racine de la structure DODAG. La notion de rang est utilisée par RPL notamment pour éviter les boucles [134].

Si ce nœud est configuré pour agir comme un routeur dans le réseau, il commence à diffuser à son tour dans son voisinage les nouvelles informations de la structure qu'il vient de rejoindre à travers des paquets DIO. Si le nœud n'est pas configuré pour être un routeur alors il rejoint tout simplement la structure DODAG et n'envoie pas de DIO.

Les nœuds voisins recevant cette annonce vont répéter ce processus de sélection de parent, d'ajout d'itinéraire et d'annonce des nouvelles informations concernant la structure DODAG à l'aide des DIOs. Ce processus continue jusqu'à couvrir tous les nœuds du réseau. Chaque nœud de la structure DODAG a une entrée de routage vers son parent (ou plusieurs parents selon la fonction d'objectif) à travers lequel ce nœud peut atteindre la racine de la structure DODAG.

Les Destination Advertisement Object (DAO)s visent à maintenir les routes descendantes et ne sont utilisés que pour des applications nécessitant des trafics de type point à multi-point et point-à-point (dire que certains sont plus courants que d'autres). Ils sont émis par les routeurs intermédiaires dans le réseau afin que les nœuds avec un rang plus petit puisse annoncer des routes pour le trafic descendant.

2.2.2.3 Tables de routage

Afin de gérer les tables de routage, une instance RPL dispose de deux modalités : avec et sans stockage des routes.

Lorsque les routes sont stockées, une table de routage doit être construite au niveau de chaque nœud. Ceci est accompli par les DAOs. Ils sont utilisés pour annoncer les nœuds qui peuvent être des destinations potentielles dans du trafic descendant. Un nœud appartenant à la structure DODAG enverra un DAO à ses parents. À la réception de ce DAO, un nœud parent ajoute une entrée dans la table de routage et il envoie à son tour un DAO à ses parents (des agrégations des informations reçues peuvent être envisagées). Ce processus se poursuit jusqu'à ce que l'information atteigne la racine du DODAG. Dans ce cas-là pour le trafic P2P les paquets remontent jusqu'à un nœud en commun puis ils sont routés depuis cet ancêtre commun.

Dans le cas où les routes ne sont pas stockées, seul la racine reçoit et traite les DAOs en provenance des nœuds et ainsi seul la racine connaît le chemin vers chaque destination. Ainsi pour atteindre un nœud, le routage est précisé explicitement par la racine. Ce cas d'utilisation peut être utile dans le cas de nœuds suffisamment contraint pour ne pas pouvoir gérer une table de routage.

2.2.3 Maintenance du routage

RPL est proactif et ne garantie pas un routage sans boucles ou des bornes sur les délais de convergence. Ce choix est acceptable dans des scénarios de smart metering dans lequel un délai supplémentaire occasionnel des paquets n'est pas critique [137] mais peut être dommageable dans des scénarios ou des délais plus stricts sont requis.

2.2.3.1 Détection et évitement des boucles

Les boucles peuvent subvenir lors de changements de topologie réseau. Elles entraînent des pertes de paquets en raison de l'expiration du Time To Live (TTL) et des congestions. RPL peut détecter et réparer les boucles lorsque du trafic est envoyé dans le réseau et qu'une incohérence est détectée. La transmission des paquets est suspendue tant que le DODAG n'est pas remis dans un état admissible ce qui peut occasionner des délais et des surcharges des mémoires tampon. Les réparations doivent être ciblées afin de ne peut pas engendrer une consommation trop importante d'énergie.

Les règles utilisées par RPL pour éviter les boucles utilisent le rang précédemment introduit. Lors de la désignation d'un parent, un nœud ne peut sélectionner qu'un voisin qui a un rang plus faible que le sien. De plus, il n'est pas possible de réduire son rang pour augmenter artificiellement le nombre de parents potentiels.

Dans le cas où RPL fonctionne avec stockage, il est possible d'utiliser les entêtes pour spécifier si le trafic est montant ou descendant. Si un routeur reçoit un paquet "descendant" et qu'il doit l'envoyer à un nœud avec un rang plus faible ou un paquet "montant" et qu'il doit l'envoyer à un nœud avec un rang plus élevé alors une incohérence est détectée et une réparation locale est déclenchée.

Dans le cas où RPL fonctionne sans stockage, la racine détecte au moment de l'envoi si un routeur apparaît plus d'une fois dans le chemin emprunté.

2.2.3.2 Réparation globale et locale

RPL dispose de deux mécanismes de réparation :

Le premier mécanisme est local, lorsqu'un nœud détecte une incohérence, il se détache du DODAG en se mettant à un rang infini ce qui "empoisonne" ses routes. Les enfants le détecte et le retire de leur liste de parents avant une reconstruction des routes.

Le second mécanisme est global. Il reconstruit l'intégralité du DODAG. La racine incrémente la version du DODAG id qu'elle propose, les nœuds propagent ce changement et reconstruisent un nouveau DODAG.

2.2.3.3 Trickle

RPL utilise un mécanisme de timer adaptatif appelé "Trickle" (goutte à goutte) afin de contrôler le débit d'émission des DIOs. Trickle double l'intervalle séparant deux émissions successives de paquets DIO à chaque fois que le réseau est cohérent, et ce jusqu'à une valeur maximale. Ainsi le trafic RPL diminue dans le réseau lorsqu'il est stable. Quand une incohérence est détectée, le timer est réinitialisé à sa valeur minimale. Un des principaux avantages de l'utilisation du timer Trickle est qu'il ne nécessite pas de code complexe et il est assez facile à mettre en œuvre. La convergence de Trickle vers le temps inter DIO maximal peut être difficile dans des conditions réelles de transmissions [120].

2.3 Fonctionnalités orientées services

La passerelle peut offrir de nombreux services. Parmi eux on peut citer l'authentification pour les objets, l'accès à des réseaux virtuels, du routage de type Interior Gateway Protocol (IGP), de la résolution de noms d'hôtes avec Domain Name System (DNS) et de la supervision réseau ou bien des services de synchronisation de temps avec Network Time Protocol (NTP). En plus des services logiciels, une passerelle peut offrir d'autres fonctionnalités de connectivité sans-fils tout en veillant à ce que les multiples radios qu'elle gère n'interfèrent pas mutuellement entre elles.

2.3.1 Reverse Proxy & Cache

Fonctionnalités de cache Un cache permet d'accélérer l'obtention d'une ressource précédemment demandée. Nous nous intéressons au Reverse Proxy Cache (RPC) utilisés pour du trafic applicatif comme sur le web [130]. Une ressource est disponible dans le cache pour être utilisée en lieu et place de la ressource réelle pendant un certain temps de validité. Le temps de validité en cache est déclaré usuellement par la ressource même. Cependant ce temps de validité peut être manipulé par le RPC qui peut prendre la décision selon le contexte d'utiliser une ressource pour un temps différent.

Spécificités liées à l'IoT Les avantages d'un RPC sont aussi importante dans le contexte de l'IoT. Un RPC peut être utile afin d'éviter qu'un nœud déjà contraint en ressources ne soient obligés de répondre plusieurs fois à une même requête. L'utilisation d'un RPC placé stratégiquement permet en outre d'éviter d'aller solliciter un réseau contraint peu rapide et permet d'avoir une latence améliorée. Dans le cas où un nœud sera accessible directement par des utilisateurs non identifiés, le RPC sera une nécessité pour contrôler les requêtes entrantes. Cette utilisation est déjà celle qui en vigueur le web [130].

2.3.2 Proxy et mise en cache

CoAP permet la mise en cache des réponses afin de répondre efficacement aux requêtes. Une mise en cache utilisant la validité des informations peut être prévue au niveau d'un nœud contraint ou d'un intermédiaire tel qu'un proxy. L'utilisation d'un proxy est utile dans les LLNs [25].

En premier lieu, elle permet de limiter le trafic et de ne pas surcharger les nœuds, les performances sont améliorées car de nombreuses transmissions sont évitées, les appareils contraints peuvent rester en veille. Enfin le proxy permet d'éviter de nombreuses attaques contre les nœuds.

2.3.3 CoAP

La couche applicative permet d'interroger les valeurs relevées par les capteurs composant le LLN. Une façon classique de gérer les données dans un réseau de capteur consiste à avoir une passerelle recevant les données depuis les capteurs puis d'envoyer ces données vers un service tiers qui sera lui interrogé par des utilisateurs finaux. Qu'il soit fait avec un protocole dédié [57] ou bien de manière ad-hoc, cette manière de procéder est particulièrement adaptée pour de la télémétrie systématique. Cette approche nécessite toute fois une étape de traduction avant de pouvoir être disponible dans des Web Services ou des services tiers.

Les requêtes de Web Services pourrait être directement envoyé sur la passerelle, qui aurait connaissance des nœuds, des ressources offertes et des protocoles utilisés dans un LLN spécifique et pourrait ainsi traduire. Cependant pour que cette approche fonctionne efficacement il est indispensable que la traduction entre les requêtes pour le service Web soit aussi proche que possible que celle a destination des noeuds du LLN. Le protocole CoAP [106] propose de combiner l'approche d'abonnement de type pub-sub avec un paradigme REpresentational State Transfer (REST) afin d'offrir une correspondance directe entre Uniform Resource Locator (URL) d'un service web et URL d'une ressource sur un capteur.

CoAP utilise l'architecture REST qui fournit une série de principes servant à construire des interfaces efficaces pour des services Web qui ont étés utilisés avec succès dans HyperText Transfer Protocol (HTTP) [103]. Ce style préconise l'absence d'état et l'utilisation de méthodes explicites pour accéder à des ressources rendues disponibles via une Uniform Resource Identifier (URI).

CoAP a plusieurs implémentations disponibles et interopérables pour systèmes contraints [64] et non-contraints [10, 65] rendant possible la création rapide de proxy comme nous le ferons dans le chapitre 4.

En dehors des choix propriétaires qui sont imposées pour certaines stack (Zigbee notamment), on peut citer Message Queuing Telemetry Transport (MQTT) [57] et CoAP.

Le choix de CoAP a été motivé pour plusieurs raisons : Son paradigme REST permet d'interfacer un LLN avec des services web classiques et permet d'avoir de manière transparente une correspondance entre service web et services offert par un LLN.

MQTT est un protocole qui utilise exclusivement un mode de fonctionnement en pub-sub mode qui est déjà pris en compte par CoAP. Ainsi nous avons jugé que les cas d'utilisation de CoAP étendaient ceux de MQTT.

Il est possible d'implémenter des fonctionnalités de traductions d'un protocole à l'autre via une couche de persistance [26].

2.3.4 Architecture REST & Observations

Ce protocole applicatif peut être vu comme une adaptation de HTTP pour les nœuds contraints et visant les applications M2M courantes.

Utiliser REST permet la traduction entre HTTP et CoAP. Cela aussi bien pour le proxy qui doit implémenter des traductions de protocoles que pour le développeur qui obtient une valeur d'une ressource comme il obtiendrait une valeur d'une Application Programming Interface (API) web. Enfin CoAP peut identifier et transporter différents formats de message tels que Extensible Markup Language (XML), Javascript Serial Object Notation (JSON) ou bien Concise Binary Object Representation (CBOR) [12].

2.3.5 Ouvertures

Il est à noter que HTTP2 supportera les notifications. Ainsi il sera possible d'avoir une implémentation complète en pub-sub tout en bénéficiant du paradigme REST.

2.4 Conclusion

Comme nous l'avons vu dans ce chapitre, les LLNs disposent d'une pile de protocoles propre et de systèmes très différents de l'Internet usuel. Cette différence est due aux contraintes énergétiques et matérielles que ces nœuds ont. Loin d'être complètement séparée des piles protocolaires classiques, cette pile se lie à l'existante via des traductions et adaptations au niveau des passerelles. Les passerelles devront avoir les deux piles afin de remplir leurs rôle. Plus généralement, les protocoles réseaux destinés aux LLNs devront coexister et être interopérable avec les réseaux classiques (Ethernet, Wifi, LTE, ...). Tous les appareils des LLNs n'utiliseront pas forcement ces protocoles et le domaine est suffisamment vaste pour qu'un écosystème de protocoles puisse se mettre en place. Tout au long des chapitres 4 et 5, nous présenterons nos contributions qui visent à adapter des services courants des serveurs classiques aux spécificités des LLNs. Nous verrons en particulier comment exploiter les informations disponibles aux différentes couches pour extraire une vue de l'activité réseau du LLN.

Cette thèse vise à montrer quelques unes de ces fonctionnalités parmi elles nous pouvons citer des fonctionnalités permettant de réguler la quantité de requêtes admise par la passerelle à destination du LLN nous proposerons une contribution allant dans ce sens dans le chapitre 4. Des fonctionnalités de supervision réseau qui seront étendues et approfondies dans le chapitre 5. Enfin une autre fonctionnalité récemment proposée est la découverte de service et leur diffusion via des systèmes pair-à-pair. Quand les nœuds vont et viennent dans le réseau, pouvoir découvrir les fonctionnalités que les nœuds offrent est essentiel. La passerelle en étant au plus près des nœuds peut faire l'inventaire des fonctionnalités que le LLN peut offrir et partager ces ressources [21].

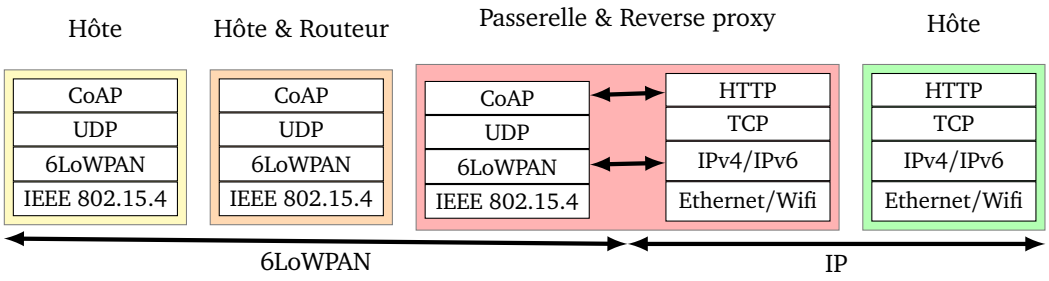


FIGURE 2.3 – Schéma de l'acheminement des requêtes dans un LLN

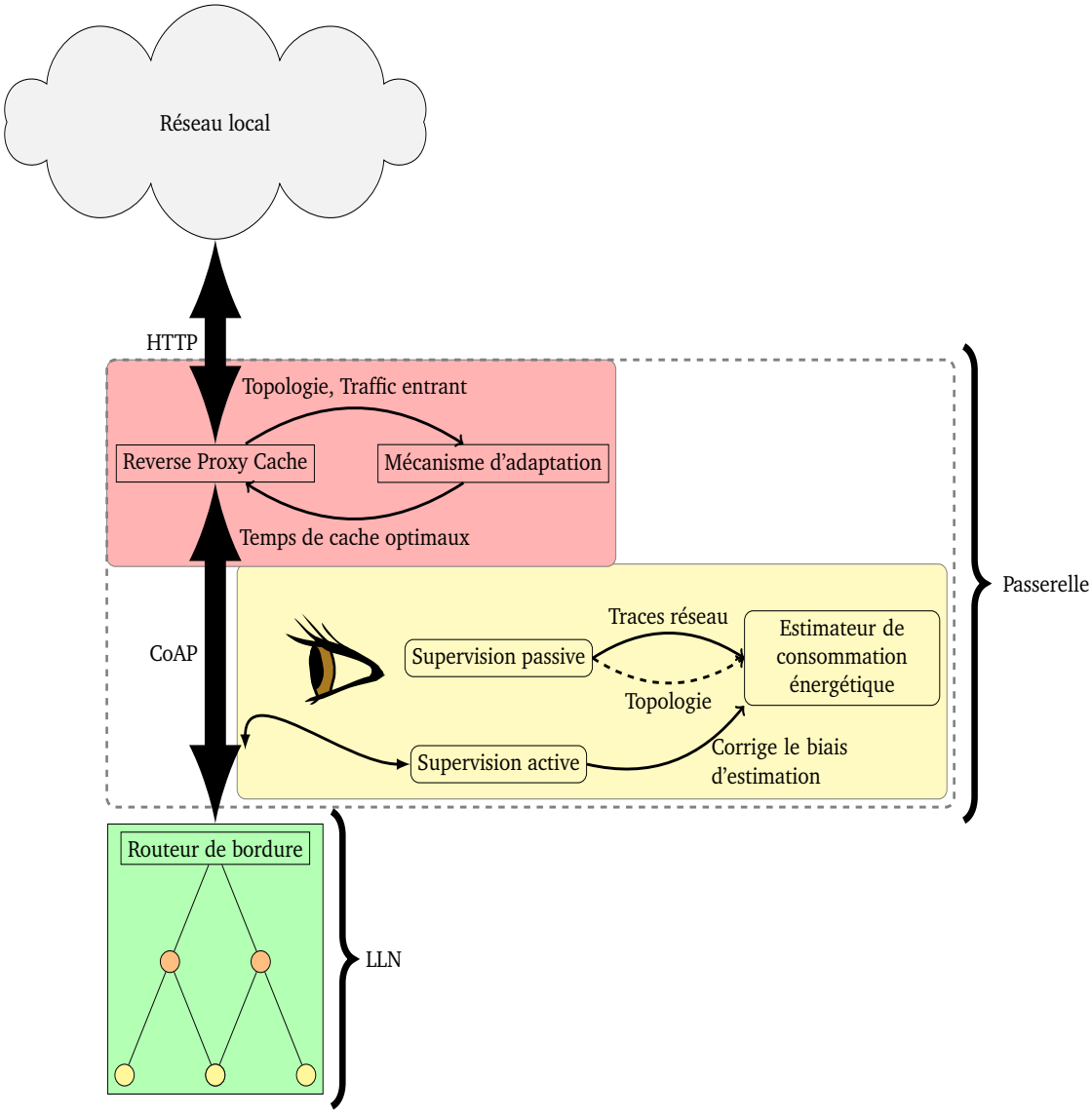


FIGURE 2.4 – Schéma de la passerelle proposée

Chapitre 3

Expériences automatisées et reproductibles pour LLNs

> It doesn't matter how beautiful your theory is, it doesn't matter how smart you are. If it doesn't agree with experiment, it's wrong.
>
> Richard Feynman

Contents

Ce chapitre présente une méthodologie et des outils pour l'orchestration, la documentation et la reproductibilité d'expériences à grande échelle. Une présentation de la reproductibilité sera donnée et mise en perspectives pour la recherche sur les LLN (3.1) puis nous présenterons notre framework et nos outils (3.1.2) et nous montrerons son application sur une expérience (3.3). Une conclusion ouvrant sur des perspectives conclurera ce chapitre (3.4). **Le lecteur peut si il le souhaite aller au chapitre 4 et 5 pour poursuivre l'étude de nos contributions sur la passerelle.** TODO : A changer pour le moment on comprends que le chapitre ne sert à rien.

Nous introduirons dans la section **??** la problématique, l'état de l'art et le positionnement de Makesense. Puis nous verrons dans la section **??** comment Makesense est utilisé pour documenter une procédure expérimentale. Nous verrons dans la section **??** comment automatiser les tâches courantes lors d'une expérience. La section **??** décrira comment Makesense rend une expérience reproductible. Enfin nous terminerons dans la section 3.4 en donnant des ouvertures sur ces travaux.

Automatiser une expérience permet de la rendre reproductible, documentée et de la partager plus facilement. Souvent jugée difficile, la reproductibilité des expériences scientifiques est pourtant une nécessité pour la démarche scientifique. Cependant mettre en place un environnement permettant de construire des expériences est long, fastidieux et n'est pas valorisé dans la communauté scientifique. Nous proposons de réduire le temps de mise en place d'expérience reproductibles en intégrant des solutions existantes venant d'autres communautés scientifiques et en les adaptant aux spécificités de la recherche sur les LLNs. Makesense est disponible sous licence Apache sur Github [1] et une démonstration du déroulement d'une expérience est également disponible [2].

3.1 Introduction à la recherche reproductible dans les LLNs

3.1.1 Répétabilité et Reproductibilité

Une expérience est définie comme une série d'actions qui a pour but de tester (confirmer ou infirmer) une hypothèse. Il y a trois éléments impliqués dans ce processus : Le *laboratoire* qui corresponds à l'environnement dans lequel l'expérience se produit, l'*expérimentateur* qui est la personne qui va faire l'expérience et le *dispositif* qui est étudié. Si une expérience peut être exécutée dans des laboratoires avec des expérimentateurs et des dispositifs tous différents mais arriver aux mêmes conclusions alors l'expérience est dite reproductible. La réplicabilité désigne le fait d'avoir exactement les mêmes résultats d'un jeu d'expérience à l'autre.

La reproductibilité est un pilier de la méthode scientifique [97, 43]. Quant à la réplicabilité, elle reste essentielle pour la vérification des résultats et la réutilisation des développements d'une expérience à l'autre. Cependant, pour l'une comme pour l'autre, elles sont rarement mises en avant, aussi bien au moment de juger un article pour le publier dans une revue ou bien après sa publication [93]. Ce paradoxe a été relevé à de nombreuses reprises dans d'autres sciences [133]. Des dépôt de documents scientifiques rendent pourtant possible l'hébergement des fichiers nécessaires au déroulement d'une expérience en plus de la publication en elle-même. Cependant cette démarche repose pour le moment essentiellement sur une volonté propre des chercheurs. Réduire le temps de développement et pour mettre en place une expérience reproductible est indispensable pour répandre ces bonnes pratiques.

3.1.2 Problématiques expérimentales des LLNs

La conception et la validation d'une expérience sur LLNs peut être réalisée par simulation ou sur des nœuds réels. Cependant, les deux approches présentent des compromis différents.

Dans le cas des simulateurs, il est possible d'effectuer une évaluation pour un très grand nombre de nœuds rapidement. Cependant, les simulateurs font un certain nombre d'hypothèses sur le médium de transmission et plus généralement sur le fonctionnement des nœuds, la dérive des horloges, l'usure des composants et les pannes éventuelles. Pour un émulateur tel que Cooja [90], il permet de rapidement tester si un micrologiciel fonctionne et si dans des hypothèses raisonnables, les réseaux se forment. Il est aussi possible de visualiser précisément les cycles de veille, la consommation

1. https://github.com/sieben/makesense
2. https://travis-ci.org/sieben/makesense

énergétique estimée et de pouvoir inspecter un nœud durant une simulation. Le contrôle sur l'environnement de simulation est total et détaillé. Si les graines des générateurs de nombre aléatoires sont fixées alors on a une reproductibilité complète de l'expérience. De ce fait, le réalisme de l'expérience est toujours modéré car les perturbations sont toujours déterministes.

Les bancs d'essai réels sont quant à eux garants d'un déploiement plus réaliste. Certaines plateformes comme Iotlab [44] offrent des traces détaillées de l'expérience, mais aucune ne peut garantir que deux expériences donneront exactement les mêmes résultats. Les architectures matérielles sont variées, les nœuds peuvent tomber en panne et les architectures et périphériques radios sont variés. Cependant, le passage à l'échelle d'expérience sur nœuds réels nécessite un effort supplémentaire pour le déploiement du code à distance, la réservation des ressources, la surveillance des erreurs survenant au cours de l'expérience et la collecte des résultats. Ainsi la mise en place de l'ensemble des outils nécessaires à la mise en place d'une expérience complète peut être long et complexe. Le temps nécessaire pour une expérience ne peut pas être accéléré et des expériences longues sont nécessaires pour tester la fiabilité des nœuds. Enfin même s'il est possible de contrôler les nœuds assez finement, il n'est pas toujours possible de contrôler l'environnement radio, la position des nœuds qui sont le plus souvent fixe et l'atténuation des signaux entre les différentes antennes.

Le cycle de développement usuel [49] représenté sur la figure 3.1, montre qu'il est assez courant de partir d'un modèle émulé ou simulé pour aller vers une implémentation réelle sur des nœuds physiques.

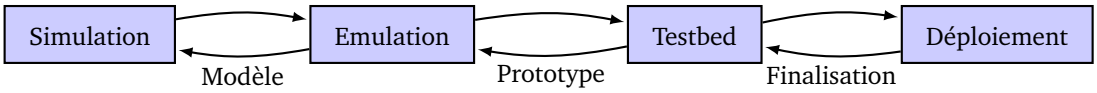


FIGURE 3.1 – Cycle de développement

3.1.3 Etat de l'art sur les outils de gestion

Les notebooks sont courants en recherche et ont été introduis par des logiciels tels que Mathematica pour améliorer les contenus pédagogiques scientifiques [138, 108]. Au lieu de créer un rapport qui soit un ensemble de fichiers disjoints ou bien un document statique, et avançant à un rythme différent de celui des résultats, on peut créer un rapport interactif qui permette de ré exécuter certaines portions de code. L'annotation et la modification des textes en marge des résultats permettent d'avoir une façon beaucoup plus lisible de partager des résultats. L'intérêt d'utiliser ce genre de bibliothèque au lieu d'une solution ad-hoc telle que Gnuplot [132] vient du simple fait qu'elle peut être intégré directement à la suite des analyses et traitements effectués sur les données de l'expérience. L'intégration est d'ores et déjà fournie à mesure que l'on travaille sur le notebook et les graphes peuvent être générés à la demande.

Jupyter s'est distingué des outils existants en étant libre et en proposant de s'interfacer avec d'autres bibliothèques scientifiques libres permettant de se poser en alternative vis à vis de solution propriétaires.

3.1.3.1 Fabrication

L'étape de fabrication prépare les fichiers qui vont être utilisés pour lancer une expérience. Cette étape de préparation compile les systèmes pour les nœuds contraints et produit tous les fichiers de configuration nécessaires à la simulation. Automatiser la fabrication des fichiers de base permet d'avoir une grande expressivité lors de la conception d'une expérience. Une approche naïve et ad-hoc

aurait du mal à fonctionner dans de grands déploiements si un grand nombre de variables différentes doivent être configurées d'un nœud à l'autre en fonction de leur position géographique par exemple.

3.1.3.2 Déploiement d'une expérience

L'interface avec les testbeds est une question étudiées [15] et de nombreux outils sont disponibles dans l'état de l'art pour s'interfacer avec des banc de tests. Cependant lors du développement de nos expériences, nous avons constaté que les hypothèses de bases utilisées par ces outils visaient des serveurs et des ordinateurs pouvant embarquer une grande quantité de code et pouvant disposer de fonctionnalités élevées. Ce n'est pas le cas des LLN qui sont des systèmes embarquées présentant des fonctionnalités assez réduites et des espaces de stockage modestes.

Nepi [67] propose l'idée d'abstraire le banc de test au profit d'objets génériques. Cependant dans le cas des nœuds des LLNs que nous déployons, leurs configurations ne leur permettent pas d'avoir toutes les abstractions d'interfaces réseaux que nepi requiert.

3.1.3.3 Exploitation de résultats

L'analyse de résultat et la production de courbes peuvent être obtenus de manière ad-hoc en utilisant des outils de filtres (grep, awk, sed) sur les fichiers en entrées et des outils de graphes (gnuplot, tableur) pour produire les courbes. Ces méthodes peuvent fonctionner dans des cas très simples, mais n'ont pas le même niveau de concision et d'expressivité dès qu'il s'agit de grouper ou d'effectuer des agrégations complexes sur plusieurs champs [101, 131].

3.2 Makesense & Documentation d'une expérience sur les LLNs

3.2.1 Présentation de Makesense et des Jupyter-notebook

Makesense [73] intègre au sein d'un même fichier l'ensemble des traitements, paramètres, fonctions et la documentation du protocole expérimental d'une expérience sur les LLN en utilisant un Jupyter-*notebook*. Jupyter (anciennement IPython [94]) est une suite d'outils libre pour l'informatique interactive et parallèle scientifique. Cette suite propose des notebooks qui combinent code, texte, images, expressions mathématiques et des fonctions interactives au sein d'un même document. Comme ce fichier texte (au format JSON) rassemble l'ensemble des paramètres, il est facile de le modifier et de l'utiliser comme modèle pour gérer d'autres expériences au sein d'une même campagne expérimentale.

Un notebook, une fois lancé, peut être vu et utilisé via un navigateur web standard comme montré sur la figure 3.2 et les cellules qui le compose peuvent être rendues ou exécutés pour fournir résultats et figures. Le résultat des exécutions des cellules de code modifie un contexte commun à l'ensemble des cellules du notebook ce qui permet de partager des variables.

3.2.1.1 Langage commun à toute l'expérience

Utiliser un seul et même langage pour organiser nos expériences et traitements est un atout. Plutôt que de chercher à coller des composants hétéroclites utilisant différents langages, nous avons privilégié une solution expressive [3] et agnostiques aux problèmes traités dans les LLNs.

Le fait de ne pas imposer de structure pour une expérience, permet de disposer d'une faible barrière d'entrée, il y a peu d'abstractions et les outils utilisés sont courants dans la communauté

3. un langage expressif permettant d'exprimer des abstractions, des fonctionnalités de haut niveau et des itérations avec peu de code.

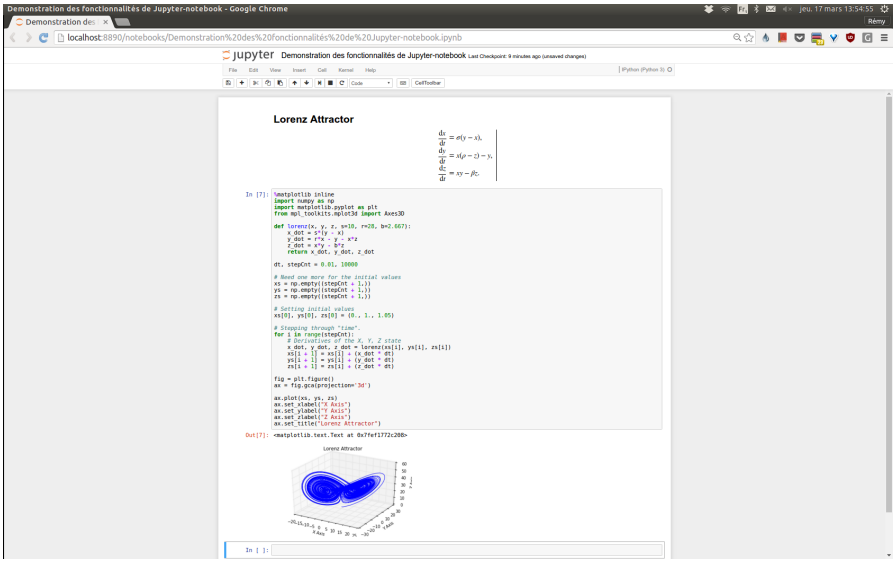


FIGURE 3.2 – Capture d'écran d'un notebook ouvert fonctionnant en local et consulté par interface web

scientifique et disposent d'une documentation abondante en plus d'être d'ores et déjà développés et maintenus.

Jupyter n'a aucun couplage entre le moteur d'exécution et le format du notebook rendant possible l'utilisation de multiples moteurs comme Python, Julia [11] ou bien R [118].

3.2.1.2 État de l'art

3.2.1.3 Partage de résultats expérimentaux

Les rapports d'expérience sont des documents enrichis et ainsi peuvent être crées de multiples façons et vers plusieurs formats textuels (HTML, PDF, ...). Grâce aux outils de conversions fournis, il est également possible de transformer le notebook en un script unique n'utilisant que les cellules de codes et qui peut être exécuté pour reproduire l'intégralité des résultats précédents. Ces outils de conversion mitigent la contrainte d'avoir un format de ce fichier commun à l'ensemble d'une équipe et permettent d'offrir un compromis intéressant en termes de choix communs et de fonctionnalités offertes.

Il est également possible d'avoir un rendu complet en ligne via Github [4] comme montré sur la figure 3.3 ou encore nbviewer [5] des cellules de textes et des images. Ainsi des collaborateurs peuvent avoir une version finale complète sans installer d'outils sur leurs machines découplant ainsi la problématique du stockage de celle du rendu dynamique.

3.2.2 Découpage en étapes et cellules

Pour Makesense, les différentes étapes d'une expérience sont découpés en cellule de notebook. Le schéma 3.4 montre comment les différentes étapes produises les entrées nécessaires aux étapes suivantes. Chacune de ces étapes se traduit en une cellule de code et peut être précédée ou suivie de plusieurs cellules de textes.

4. https://github.com/sieben/makesense/blob/master/demo.ipynb
5. http://nbviewer.jupyter.org

FIGURE 3.3 – Capture d'écran d'un notebook ouvert

3.2.2.1 Cellule de texte

Lorsqu'une cellule de texte riche est rendue, le texte et les formules mathématiques au format LaTeX ou Markdown contenu dans cette cellule sont rendus à l'utilisateur. Ainsi il est possible d'avoir une mise en page avec des listes, des sections, des images et des formules mathématiques au sein même du notebook.

3.2.2.2 Cellule de code

Lorsqu'une cellule de code est rendue, le code qui est dans la cellule est exécuté dans le noyau qui est commun à l'ensemble du notebook. Le code executé modifie le noyau qui est commun à l'ensemble des cellules et les fonctions et variables sont ainsi partagées ce qui permet d'avoir une grande interactivité. De plus, les cellules de code peuvent être ré-executées indépendamment les unes des autres. Documenter chaque cellule de code avec du texte riche et des images rend le processus expérimental beaucoup plus clair et interactif pour des collaborateurs qui ne sont pas directement impliqués

Ces cellules de code seront celles qui seront extraites dans le cas d'une conversion du notebook vers un script executable.

3.2.3 Intégration Continue

L'intégration continue est un ensemble de pratiques utilisées en production de logiciel qui consiste à vérifier que chaque modification apportée à un programme ne modifie pas son fonctionnement et qu'aucune régression fonctionnelle n'est introduite [36]. Son principal but est de détecter les problèmes et les erreurs afin de les corriger au plus tôt et de raccourcir les cycles de développement. Des logiciels en licence libre tel que Jenkins [114] ou bien des plateformes intégrées telles que Travis-ci [95] offrent des fonctionnalités d'intégration continue.

Les notebooks que nous avons présentés peuvent être transformés en script et exécutés pour savoir si l'expérience qu'ils décrivent fonctionne ou non en fonction du résultat du script. L'intégration

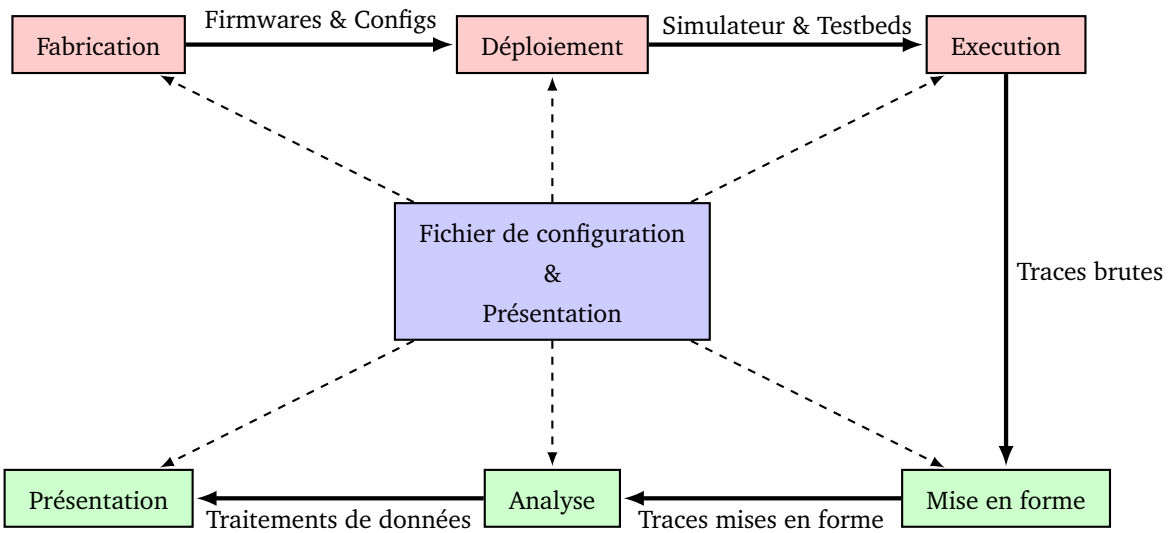


FIGURE 3.4 – Découpage des étapes dans Makesense

continue est déclenchée quand une proposition de changement (“pull request”) est poussée vers le dépôt gérant les versions de notre code. Une copie des changements est envoyée sur le serveur d'intégration continue qui va créer l'environnement pour le notebook puis exécuter les traitements qu'il contient. Une fois que le résultat est conforme aux tests demandés il peut être intégré à la branche principale de développement et le processus se répète pour tout nouveau changement proposé. Ainsi on a la garantie qu'une modification ne casse pas le flot de l'expérience ni ses résultats.

Nous avons utilisé Travis-ci [46] comme serveur d'intégration continue en raison de la disponibilité de configurations pour l'environnement Contiki [33] et de son intégration à Github qui héberge nos notebooks et propose un rendu direct en ligne. Travis-ci étant une structure indépendante et ouverte, elle constitue un bon gage sur le fait que n'importe qui pourrait refaire l'expérience et s'assurer de nos résultats. Ainsi nous avons pu construire une démonstration présentant les différentes étapes présentées dans ce chapitre et les avons exécutées sur cette plateforme [72].

3.3 Automatisation d'une expérience sur les LLNs

Automatiser les tâches d'un processus est essentiel pour s'assurer qu'aucune étape n'a été oubliée ou bien pour expliciter l'ensemble des étapes faites. De nombreux outils ont été développés pour faciliter l'écriture de tâches dépendants les unes des autres [40]. Makesense automatise une expérience en exécutant le contenu des cellules de codes contenues dans le notebook.

Afin d'illustrer l'intérêt de Makesense, nous allons exposer comment automatiser une expérience utilisant Contiki [32] et le simulateur Cooja [90].

3.3.1 Fabrication

Dans le cas du code exécuté par les nœuds, il peut être obtenus pour différentes architectures [6].

6. MSP430 via les cibles wismote ou sky dans le cas des simulations COOJA et ARM dans le cas des Cortex M3 présent sur IoTlab

Une simulation dans Cooja requiert la création d'un fichier principal qui va être utilisé pour placer les nœuds, les démarrer et interagir avec eux, configurer leur portée de transmission ou la graine des générateurs de nombres aléatoires utilisée. Ainsi la génération des positions des nœuds est rendue automatiquement et les code nécessaires sont affectés aux nœuds correspondants. Un cas d'usage classique de cette étape est de créer une campagne de simulations avec des nœuds ayant des fonctionnalités et différents paramètres appliqués individuellement. Dans le cas de nœuds très contraints, il est possible de générer un code source et de diminuer la gestion dynamique des arguments que nous souhaiterions passer au nœuds.

3.3.2 Déploiement - Exécution - Déplacement des traces

Déployer sur un banc de tests mets en jeu de nombreux éléments. Il faut en premier lieu effectuer une réservation sur les nœuds que l'on vise, maintenir une table de correspondance entre les nœuds réservés et le code que l'on souhaite exécuter sur ces nœuds. Automatiser le processus de déploiement permet d'avoir une génération dynamique de ces correspondances et permet de changer rapidement les nœuds visés en fonction des pannes et indisponibilités du banc de test visé.

Que l'exécution se passe dans un simulateur ou sur un banc de test, il est courant de gérer plusieurs processus en parallèle lors d'une exécution. Ainsi il est nécessaire de gérer l'injection du trafic, le maintien des canaux de communications (Tunnels, Interfaces virtuelles), l'agrégation des messages étant envoyés sur le port série, etc... Ainsi il devient compliqué de gérer l'ensemble de ces processus et de redémarrer ceux rencontrant des erreurs. Une fois terminée les traces de l'expérience doivent être récupérées pour être exploitée en local. Ainsi il est utile d'automatiser tous ces processus afin de ne rien oublier à chaque lancement.

3.3.2.1 Méthodes & outils utilisés

Fabric [45] est une bibliothèque permettant de transmettre et d'exécuter des programmes sur des serveurs distants. Depuis la ligne de commande, il va être possible de lancer différentes fonctions codées en Python. Dans le cas du banc de test Iotlab [44], l'envoi et la récupération des différentes traces ont été automatisé afin de pouvoir reproduire n'importe quelle expérience lancée.

3.3.2.2 Comparaisons avec les autres méthodes / État de l'art

Utiliser un écosystème de logiciels ayant des communautés larges et ancrées dans les problématiques scientifiques est un choix que nous jugeons plus pertinent à celui de construire un Domain Specific Language (DSL) pour résoudre un seul type de problème ce qui restreint d'office son impact.

Makesense n'introduit aucune interface pour l'utilisation d'un testbed. Ainsi, l'interopérabilité entre les différents testbeds ne peut être garantis que si ces testbeds sont eux-mêmes interopérables entre eux. Dans le cas de la plateforme FIT-IoT lab, elle présente la même interface sur plusieurs sites différents et makesense fonctionne sur l'ensemble de ces sites puisqu'il utilise la bibliothèque fournie pour interagir avec leurs plateforme.

3.3.3 Mise en forme des résultats bruts

Une fois que l'exécution de l'expérience est terminée, il est nécessaire de transformer les résultats bruts qui en sont issus vers des données plus exploitables. Dans le cas de nos expériences, il s'agit de transformer les fichiers Packet CAPture (PCAP) et les multiples journaux textes représentant les événements qui se sont produits. Cette étape étant dissociée des contraintes et du contexte des LLNs, tous les traitements que nous ferons peuvent utiliser des outils généraux et communs à de nombreux

champs de recherche. Au vu des volumétries mises en jeu lors de nos expériences nous avons préféré rester sur des solutions de traitements sur des fichiers Comma Separated Values (CSV) pour leur facilité d'utilisation.

Certaines transformations peuvent être triviales comme des conversions d'unités et des normalisations de valeurs. D'autres peuvent être la transformation des adresses Media access control (MAC) vers des identifiants plus facilement compréhensibles et pouvant nous aider à faire une cartographie géographique des événements dans le réseau. Enfin, dans le cas de système très contraints, des sorties textes compactes sont remplacés vers des noms plus détaillés. Dans le cas d'un trafic réseau, il est nécessaire de pouvoir classifier chaque paquet selon une série de critères qui vont nous permettre dans la prochaine phase d'analyse d'effectuer des traitements quantifiés. Une fois que toutes les transformations sont faites, nous avons à notre disposition un format de données CSV standardisé que nous allons pouvoir analyser en détail.

Il serait possible d'utiliser des outils ad-hoc pour faire de l'exploration des traces sans aucune automatisation. Cependant cette approche serait difficilement documentable et partageable à mesure que le volume de traitement à faire augmente. L'automatisation apporte une expressivité à l'ensemble des traitements effectués. De plus, transformer les données vers CSV permet de les importer dans de nombreux outils (Pandas, Excel, ...) plus facilement notamment dans le cas de données tabulaires.

3.3.4 Analyse des résultats

La phase d'analyse de résultats est en charge d'effectuer des traitements sur des données misent en forme. C'est lors de la phase d'analyse que va se faire la recherche et la production de résultats qualitatifs et quantitatifs sur une expérience. L'analyse des résultats doit être facilité par des cycles d'itération aussi court que possible pour relancer des expériences, tester de nouvelles hypothèses et avoir des arguments quantifiables pour prendre des décisions rapides. L'intégration des outils de visualisation instantanée directement dans le notebook rendent l'exploration des données aisée et la production de courbes est immédiate permettant de partager graphiquement, analyse des données, code et résultats avec d'autres personnes.

Dans le cas de nos expériences sur les LLNs, nous avons voulu avoir une représentation du DODAG formé par les noeuds. Nous avons d'abord rassemblé toutes les préférences de parents pour les nœuds du réseau, puis nous avons injecté ces liens dans une bibliothèque de manipulation et de visualisation de graphe (NetworkX). Cela nous a permis d'avoir une représentation graphique du DODAG RPL, de calculer les plus courts chemins entre un nœud et la racine du DODAG pour calculer une profondeur d'un nœud.

Nous avons aussi géré l'analyse de données tabulaires telles que celles obtenues en analysant les paquets émis au cours de l'expérience. Pour cela nous avons utilisé Pandas [81] qui est une bibliothèque Python permettant la manipulation et l'analyse des données tabulaires et de séries temporelles. Cette bibliothèque permet la visualisation directe dans le notebook de grande quantité de données rangées en tableau et indexées.

Par exemple l'analyse de la répartition de protocoles réseau est faite en sélectionnant l'ensemble des paquets puis en les groupant sur le champ décrivant le protocole utilisé. Ces traitements sont proches de ceux qui seraient faits en Structured Query Language (SQL) permettant ainsi un formalisme efficace pour le traitement de ces données. En plus de la visualisation instantanée, Pandas permet aussi afin de raffiner encore plus une analyse en chaînant les traitements les uns derrière les autres. Ces fonctionnalités de groupement d'informations permettent d'exprimer de manière concise des traitements sur les données pour en extraire les informations utiles comme montré dans l'annexe B.4.

3.4 Conclusion & Perspectives

TODO : Dire qu'on aimerait gérer les erreurs à la volée, déconnexions, re déploiement du code des motes, la supervision et la prise de décision pendant qu'une expérience se déroule mais c'est délicat car les pannes peuvent être très nombreuses et les décisions à prendre peuvent dépendre de nombreuses variables.

TODO : Travis-CI a changé les fonctionnalités au cours de la rédaction de ce manuscrit invalidant une partie des décisions prises au départ pour les expériences nécessitant des interfaces virtuelles et l'injection de trafic venant de l'extérieur. Ainsi nos expériences sont complètement documentée mais pas complètement intégrés sur cette plate-forme aujourd'hui dans des scénarios spécifiques. Des scénarios auto-hébergés nous donnant un contrôle sans entraves seraient aujourd'hui choisis.

Makesense propose de documenter et d'automatiser une expérience pour LLN transcrite dans un notebook pour s'assurer de sa réplicabilité. La reproductibilité d'une expérience doit devenir une problématique de premier plan car l'investissement en temps pour la mettre en place ne peut être justifié que si la reproductibilité est une nécessité des projets de recherche. La réduction de la complexité de la mise en place d'une expérience permet d'accélérer une étude et d'augmenter son impact en facilitant sa vérification et son partage.

Les meilleures pratiques de développement logiciel influencent les méthodologies expérimentales notamment en informatique. La création de communautés autour des différents outils logiciels facilite les échanges et la diffusion de nouvelles méthodes plus efficaces. Dans le cas des LLNs, l'intégration de multiples architectures matérielles et les matrices de tests qui en découle rendent cette approche encore plus pertinente.

Publications

- Rémy Leone, Jeremie Leguay, Paolo Medagliani, Claude Chaudet, et al. Makesense : Managing reproducible wsns experiments. *Fifth Workshop on Real-World Wireless Sensor Networks*, 2013.
- Rémy Léone, Jérémie Leguay, Paolo Medagliani, and Claude Chaudet. Demo abstract : Makesense—managing reproducible wsns experiments. In *Real-World Wireless Sensor Networks*, pages 65–71. Springer, 2014.
- Rémy Léone, Jérémie Leguay, Paolo Medagliani, and Claude Chaudet. Demo Abstract : Automating WSN experiments and simulations. In *EWSN*, 2015.

Chapitre 4

Optimisation des ressources d'un LLN avec un cache intelligent

> There are only two hard problems in Computer Science : cache invalidation and naming things.
>
> Phil Karlton

Contents

Nous nous intéresserons dans ce chapitre à l'utilisation d'un RPC au niveau de la passerelle afin d'améliorer les performances d'un LLN. Nous introduisons et présenterons l'état de l'art, l'architecture choisie et une justification de notre approche (4.1). nous effectuons une simulation avec un RPC pour quantifier les gains et avoir une validation préliminaire de notre système (4.2.1). Puis, nous verrons comme faire un RPCA qui adapte les temps de validité des URI en fonction d'objectifs concurrents

(4.4) via une approche multi-objectifs qui sera détaillée (4.4.2). Enfin nous conclurons ce volet en présentant quelques ouvertures de nos travaux (4.5).

Les requêtes applicatives venues de l'extérieur peuvent être ralenties par les conditions difficiles de transmissions et un faible débit. De plus les problèmes de congestion et de pertes de paquets justifient la mise en place de solution qui limite autant que possible la sollicitation du LLN. Lorsque plusieurs clients souhaitent consulter la même information, un RPC permet également d'éviter les communications redondantes. Cependant configurer un RPC efficacement est un défi dans les LLNs où plusieurs métriques contraires doivent être optimisées simultanément.

Bien vérifié qu'il y a une vraie précision dans l'utilisation des termes passerelle, routeur, reverse proxy, cache et reverse proxy cache adaptatif (ma contribution). Ces mots ne sont pas synonymes. Utilisez autant que faire se peut RPC et RPCA

4.1 Introduction

La passerelle entre un LLN et un réseau local peut contenir de multiples services tels qu'un pare-feu, de l'agrégation ou de la compression [54]. Ces services peuvent s'appuyer sur la connaissance de la topologie du réseau, des services offerts par les nœuds du LLN, les applications extérieures qui les utilisent, les paramètres de la couche MAC, l'énergie résiduelle des nœuds. L'utilisation d'un RPC afin d'améliorer la réactivité d'un système et réduire son utilisation est une pratique courante [51] sur de nombreux sites Web à fort traffic. La mise en cache consiste à mettre la réponse d'une requête directement accessible au niveau de la passerelle afin de la servir comme réponse à une requête éventuelle sans solliciter de nouveau le serveur dont elle est issue. Ainsi des gains de rapidité et d'économie de la bande passante sont observés. Dans les approches classiques, les configurations des RPC ne dépendent pas de l'état des serveurs qu'il gère. Cependant dans le cas des LLN des gains de performances et des économies d'énergie sont à réaliser quand une vue globale du système est disponible pour faire des choix de configurations judicieux.

4.1.1 Motivations pour l'utilisation d'un RPC pour les LLNs

4.1.1.1 Réduction de latence et de bande passante consommée

Dans un contexte contraint, l'intérêt des mécanismes de mise en cache est encore plus important. Une des principales raisons de développer une architecture de mise en cache pour les LLNs est de permettre aux noeuds d'économiser de l'énergie par la mise en cache d'informations qui ont déjà été demandées par un client distant. De cette façon, les nœuds peuvent continuer de rester dans un état de sommeil puisque les requêtes venant de l'extérieur sont directement servies par le RPC. De plus, cela améliore également le délai de réponse à une requête entrante. En effet, la bande passante et les conditions de transmissions d'un LLN étant restreinte, envoyer une requête prends un temps non négligeable pour une information qui peut être encore valide.

4.1.1.2 Traductions protocolaires : Sémantique & Implémentations

Comme nous l'avons vu dans le chapitre 2, il peut être pertinent pour le reverse proxy hébergé sur la passerelle de pouvoir gérer plusieurs protocoles. Ainsi disposer du contenu des ressources permet de les rendre disponibles par plusieurs protocoles différents. Dans le cas de HTTP et CoAP, la traduction d'un protocole a l'autre est facilitée par l'utilisation du paradigme REST.

En plus de cette sémantique, d'autres options telles que les Entity Tag (ETag) sont communes. Elles permettent à CoAP dans le cas où la réponse d'une requête à expiré de gérer la revalidation

d'une réponse. Ainsi dans le cas où la réponse à une requête est toujours la même, elle est juste reconfirmée et non transmise ce qui permet d'avoir une réponse de revalidation plus compacte.

4.1.1.3 Contexte pub-sub

Comme nous l'avons vu dans la section 2.3, les mécanismes d'observations sont pris en charge par les protocoles CoAP et même pour HTTP2. Ainsi on peut disposer d'une sémantique de type REST tout en bénéficiant de ce paradigme adapté au scénario de télémétrie.

Dans le cas de demandes d'abonnement émises par un client distant, le RPC les traite par le maintien d'une liste de ressources observées et une liste de clients abonnés. Chaque fois qu'une notification de mise à jour de ressource est envoyé par le nœud vers le reverse proxy, le reverse proxy transmet ces mise à jour aux abonnés correspondants par les protocoles adaptés. En outre ces informations sont disponibles au cas où des requêtes simples et sans demande d'abonnement arriverait.

Lorsque l'envoi des messages de la part des noeuds est déterministe, il est possible pour le RPCA d'inférer des profils de traffic et de s'assurer qu'ils sont conformes aux consignes des administrateurs. Ainsi en cas de changements de configurations réseaux ou des consignes des administrateurs, le rythme de mise à jour des observations sont adaptées.

4.1.2 Contribution

Nous confirmerons dans un premier temps l'impact de la configuration d'un RPC sur un LLN et en particulier sur sa durée de vie. Puis nous verrons comment obtenir une durée de vie aussi longue que possible quand d'autres paramètres telle que la satisfaction des utilisateurs rentre en jeu. Sans aucune autre contrainte, le reverse proxy devrait mettre les durées de cache au maximum pour augmenter la durée de vie. Nous montrerons qu'en modélisant le problème comme un problème multiobjectif où la satisfaction des utilisateur rentre en ligne de compte afin de former un RPCA où les temps de validité d'une information en cache sont calculées à la volée.

4.1.3 État de l'art

Un RPC se tient entre un ou plusieurs serveurs et un ou plusieurs clients. Il intercepte les requêtes et réponses en sauvegardant des copies des réponses aux requêtes. Il peut s'agir de pages complètes ou bien de fichiers plus volumineux comme des images ou des résultats de traitements longs. Ainsi, si une nouvelle requête se présente pour la même adresse URI, le RPC pourra la servir sans solliciter les serveurs originaux [48]. Aujourd'hui une grande partie des sites web les plus consultés du monde utilisent ce type de service pour alléger la charge sur leurs serveurs principaux [102].

4.1.3.1 RPC pour les LLNs

De nombreux travaux académiques [24, 25] s'appuient sur les spécifications de CoAP pour produire des architectures interopérables avec des protocoles existants. Cependant les aspects énergétiques ne sont pas directement présents au cœur des réglages de ces reverse proxys. L'utilisation de cache au sein même des réseaux est également une technique explorée [35], cependant les paramètres de gestion des temps de validité dans un cache sont le plus souvent réglés à une valeur constante et indépendante de l'état du réseau.

Notre approche consiste à gérer les paramètres de ces interfaces via des modèles et des informations disponibles au niveau de la passerelle afin de gérer au mieux ce reverse proxy pour qu'il puisse réduire la latence d'une requête, éviter les congestions superflues dans le réseau et améliorer sa durée de vie.

4.1.3.2 Reverse proxy cache adaptatif

En HTTP, calculer de manière heuristique des durées de vie de cache est une technique connue [83, 48]. Cependant les heuristiques employées se basent essentiellement sur les dates de modifications d'une URI. Ces temps de validité ne sont mis que dans des cas où une option d'age maximal de validité n'est pas déjà présente. En outre, une alerte dans les entêtes HTTP prévient les utilisateurs finaux qu'une heuristique est employée dans le traitement de leur requête si celle ci dépasse une certaine durée (fixée à 24heures dans le cas de HTTP [42]).

Notre approche vise à adapter cette heuristique en fonction de la topologie réseau sous jacente et de la consommation énergétique en vue d'obtenir une durée de vie précise.

4.2 Architecture d'un RPCA pour LLN

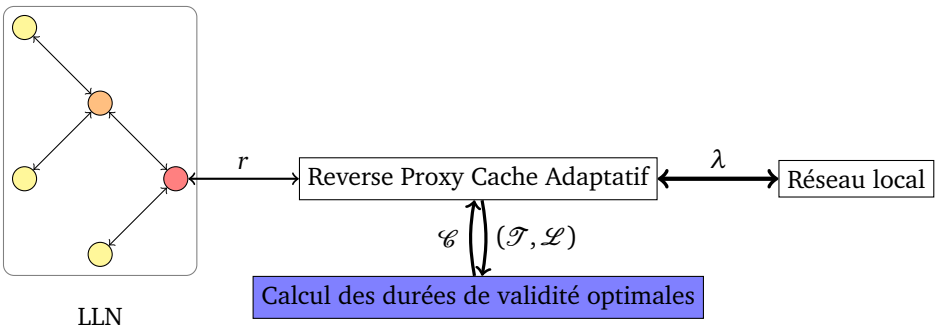


FIGURE 4.1 – Architecture du RPCA

Comme présenté dans la Figure 4.1, le LLN est composé d'un ensemble de nœuds connectés entre eux formant un DODAG ayant pour racine un routeur de bordure. La passerelle reçoit des requêtes provenant du réseau local pour chaque URI. Soit λ_u le nombre moyen de requêtes reçus à destination de l'URI u. Le RPC agit sur les requêtes entrantes et cela produit une quantité r_u de requête qui sont transmises à travers le LLN vers le nœud hébergeant l'URI u. Par simplicité d'écriture nous appellerons λ et r l'ensemble des λ_u et r_u respectivement. Il est à noter que dans cette architecture, le reverse proxy est celui qui transmet la requête entrante au LLN. Ainsi le fait de n'avoir qu'un seul cache pour plusieurs routeurs de bordure permet d'éviter les problèmes liés à des caches désynchronisés.

La passerelle implémente un mécanisme de cache lui permettant de garder en mémoire les résultats des requêtes passées pendant une durée de vie c_u. Une requête visant une URI u, venant de l'extérieur, en cache depuis moins de c_u sera traitée par la passerelle sans solliciter le LLN. Étant critiques, ces paramètres c_u doivent être choisis avec précision. Un c_u trop petit rendra la mémoire cache inefficace car invalidée rapidement. À l'inverse c_u trop long risquera de donner des valeurs obsolètes.

Pour effectuer son calcul, le RPCA requiert la connaissance de la topologie réseau $\mathscr{T}$, de la durée de vie $\mathscr{L}$. Une solution $\mathscr{C}$ contenant l'ensemble des temps de validité c_u va être ensuite déployée sur le RPCA. Bien évidemment, le choix du $\mathscr{C}$ est dynamique et peut être recalculé au cours du temps en fonction de l'évolution du LLN.

4.2.1 Performance d'un reverse proxy cache

Dans un premier temps, nous verrons l'impact qu'un RPC peut avoir sur la quantité de requêtes gérée par un LLN. Ainsi nous pourrons voir que les intérêts immédiats du cache comprennent une

réduction des délais de réponse et une supervision facilitée de notre LLN vis-à-vis des notifications et de la topologie.

4.2.2 Modèle théorique

Nous modélisons le taux d'arrivée de chaque requête reçu par l'URI u ressource est distribué comme un processus de Poisson de paramètre λ_u.

Si le reverse proxy a une valeur stockée qui est encore valide, qui est dont la durée de vie est inférieure à une valeur donnée c_u, il répond directement à la demande d'un client. Sinon, si la valeur requise est pas présent ou il est plus âgé que c_u, il transfère la demande au serveur hébergeant cette URI.

Soit Cache Miss ratio (m) la probabilité qu'une requête ne puisse pas être satisfaite par le RPC. Ainsi, m correspond à la probabilité que le temps entre chaque arrivée d'une requête soit plus grande que c_u :

$$m = \int_{c_u}^{\infty} \frac{e^{-\frac{t}{T}}}{T} \mathrm{d}t = e^{-\frac{c_u}{T}}. \tag{4.1}$$

Ainsi on déduit Cache Hit ratio (h) qui est le complémentaire de m :

$$h = 1 - e^{-\frac{c_i}{T}} \tag{4.2}$$

Le temps d'inter arrivées T_u (dimension : [s]) entre deux requêtes consécutives peut être modélisé comme un variable aléatoire exponentielle de paramètre λ_u. Étant donné que la demande pour la ressource u arrive en moyenne chaque T_u, il peut être prouvé que la durée moyenne r_u (dimension : [s]) entre deux demandes consécutives peut être défini comme :

$$r_i = \begin{cases} \lceil \frac{c_i}{T_i} \rceil T_i & \text{si } T_i \leq c_i \\ T_i & \text{sinon.} \end{cases} \tag{4.3}$$

Ainsi dans le cas où plusieurs URI u sont gérées par un même noeud i on se retrouve avec une fréquence par noeuds qui vaut :

$$r_i = \sum_{u \in i} \frac{1}{c_u} \tag{4.4}$$

4.2.2.1 Bornes pour les durées de validité

Pour chaque URI que le RPC doit gérer, plusieurs informations sont définies :

$c_{max}(u)$ qui est définie comme la durée maximale que l'administrateur du réseau que l'administrateur réseau est prêt à garder cette information en cache. Réciproquement on définie $c_{min}(u)$ qui est la durée minimale qu'il veut garder en cache. Ainsi le rôle du RPC consiste à choisir une durée de vie en cache $c(u)$ défini comme :

$$c_{min}(u) \leq c_u \leq c_{max}(u)$$

En outre, nous appelons $\mathscr{C}$ l'ensemble des paramètres $c(u)$ pour toutes les URI u que le RPC connaît.

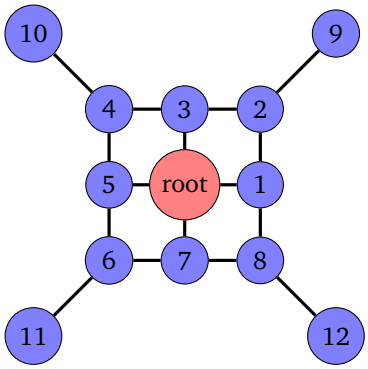


FIGURE 4.2 – La topologie radio considérée. $N = 12$ nœud placés sur une grille avec la racine comme nœud central.

Débit de transmission	R	250 kbps
Intervalle entre deux tentatives de préambules	T_p	0.4 ms
Temps pour détecter un paquet Acknowledgement message (ACK)	T_d	0.16 ms
Taille d'une requête (`GET`)	L_r	87 octets
Taille d'un paquet de réponse	L_a	96 bytes
Temps pour transmettre un IEEE 802.15.4 ACK	$T_{\mathscr{A}}$	0.608 ms
Puissance consommée lors de la transmission	$P_{\mathscr{T}}$	0.0511 W
Puissance consommée lors de la réception	$P_{\mathscr{R}}$	0.0588 W
Puissance consommée lors du sommeil	$P_{\mathscr{S}}$	$2.4 \cdot 10^{-7}$ W
Nombre de nœuds dans le LLN	N	12
Nombre de run par configuration		10
Nombre de requêtes CoAP traité par noeuds		50

TABLE 4.1 – Paramètres utilisés dans les simulations.

4.2.3 Scénario de la simulation

Afin d'évaluer le modèle de serveur cache introduit précédemment, nous effectuons la simulation suivante.

12 serveurs CoAP utilisant Contiki comme système d'exploitation sont déployés et émulés via Cooja. Afin d'avoir une empreinte mémoire faible, chaque serveur n'a qu'une seule ressource et afin de faciliter l'analyse sera un texte de taille fixe ne causant pas de fragmentations. Nous prenons pour hypothèse de travail que les capacité de mémoire pour la passerelle sont suffisante pour faire retenir toutes les réponses pour chaque URI disponible dans le réseau. Dans nos expériences, nous avons pour hypothèse que chaque nœud du LLN ne gère qu'une URI.

La figure 4.1 représente la topologie radio utilisée où un lien signifie que les nœuds peuvent être en communication radio l'un avec l'autre.

Les paramètres principaux de la modélisation du système sont listés dans la table 4.1.

Les paramètres de consommation d'énergie présentés dans ce tableau ont été prises à partir de la fiche produit interne d'un nœud prototype. Comme les autres nœuds de capteurs commerciaux

bien connus utilisant le chipcon de CC2420 (par exemple, Arbalète MicaZ, Berkeley Telosb [96]), la consommation d'énergie est plus élevé dans le mode de réception que dans le mode de transmission à pleine puissance.

Les demandes du client distant sont interceptés par le reverse proxy, qui traduit les requêtes HTTP en CoAP et, inversement, traduit les réponses CoAP réponses en réponses HTTP. En outre, le reverse proxy met en cache les réponses rendues par les nœuds afin de les mettre à disposition pour une éventuelle autre demande entrante.

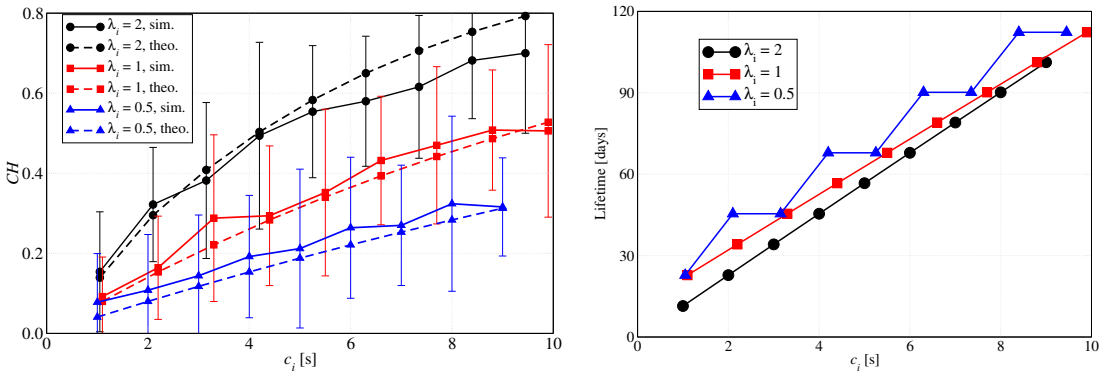
Afin d'évaluer les performances de notre implémentation, nous évaluons d'abord le cache hit ratio h comme une fonction de la temps de validité $\mathscr{C}$. Nous indiquons pour chaque courbe l'intervalle de confiance à 2σ, où σ est la déviation standard sur plusieurs exécutions successives.

Nous définissons la durée de vie d'un réseau comme l'intervalle de temps entre son démarrage et la perte de son premier nœud à court d'énergie et que tous les nœuds démarrent avec la même énergie initiale.

Le réseau que nous utilisons est composé de 12 nœuds fixes connectés via un DODAG. Les nœuds utilisent ContikiMAC comme mécanisme de cycle de veille. Le traffic CoAP démarre lorsque le DODAG de RPL a convergé. Nous prenons pour hypothèse que la topologie du réseau est statique pendant le temps de traitement de la requête, que les pertes de paquets sont négligeables car le réseau est en hypothèse de trafic faible. Le temps entre chaque requête est modélisé par un temps d'attente exponentiel de paramètre λ_u et que $\forall i, \lambda_u = \lambda$. Nous calculons la durée de vie du réseau en utilisant la topologie et une modélisation de ContikiMAC qui est la couche MAC que nous avons utilisé et qui est présentée dans l'annexe A. Ainsi notre approche est compatible avec n'importe quel autre couche MAC dont la modélisation fournit les mêmes paramètres en sortie.

4.3 Validation expérimentale

4.3.1 Résultats expérimentaux pour un trafic de Poisson constant



(a) (b)

FIGURE 4.3 – Analyse du hit ratio et de la durée de vie

(a) h comme fonction de la durée de vie de cache c_u. Simulation (traits plein) et théorique (pointillés). (b) Durée de vie du LLN comme fonction de la durée de vie dans le cache c_u. $\lambda = 2$ (cercles), $\lambda = 1$ (carrés), et $\lambda = 0.5$ (triangles).

La simulation (traits pleins) et théoriques (lignes en pointillés) sont représentées pour différentes valeurs de paramètres λ. Toutes les courbes de la figure 4.3 (a) croissent quand le paramètre c

augmente. Il est clair que plus les temps de validité dans le RPC sont élevées, plus la probabilité que la requête soit servie par le RPC augmente épargnant ainsi aux nœuds du LLN de les traiter. L'effet bénéfique de l'utilisation de la mise en cache ne se limite pas à l'énergie économisées liées à la diminution du nombre de paquets transférés à la LLN avec l'utilisation de proxy. La durée de vie du réseau est amélioré comme le montre la figure 4.3.

La forme en escalier de la courbe avec $\lambda_i = 0.5$ est due à la définition de r_i dans (4.3) qui implique que différentes valeurs de c_i conduisent au même nombre de requêtes transmises à la LLN.

Ainsi si $\mathscr{C}_{min}$ garantie une fraicheur aussi élevé que possible mais raccourcit la durée de vie alors que $\mathscr{C}_{max}$ garantie une durée de vie aussi longue que possible. Une valeur haute de c_u sera utilisée pour des nœuds ayant peu d'énergie ou avec des informations peu changeante. Par contre, une valeur c_u petite sera pour des informations devant être aussi récentes que possible.

4.4 Reverse Proxy Cache Adaptatif

Dans le cas où le seul objectif est la durée de vie du réseau, toutes les durées de temps de validité des réponses doivent être mises au maximum admissible c'est à dire : $\mathscr{C} = \mathscr{C}_{max} = c_{max}(i), \forall i$. Cependant cette durée de vie peut être surdimensionné par rapport aux besoins de l'application au détriment de la fraicheur des réponses. Nous étendrons les fonctionnalités du RPC en ajoutant un mécanisme de calcul de l'ensemble des solutions optimales $\mathscr{C}$. Toutes ces configurations forment un front non-dominés de Pareto qui sont solution de ce problème multi-objectifs. Le but est de fournir au RPC un ensemble de solutions admissibles et de trouver celle qui offre le meilleur compromis entre la satisfaction d'un utilisateur et la durée de vie du réseau.

4.4.1 Satisfaction d'un utilisateur

Pour notre modélisation nous allons considérer que plus une information donnée à l'utilisateur est récente, plus ce dernier est satisfait. Ainsi nous introduisons γ_u mesurant la satisfaction d'un utilisateur par une variable normalisée et linéaire :

$$\forall u, \gamma_u = \frac{c_{max}(u) - c_u}{c_{max}(u) - c_{min}(u)} \tag{4.5}$$

En particulier, une satisfaction est minimale quand $c_u = c_{max}$ et une satisfaction est maximale quand $c_u = c_{min}$. Nous obtenons ainsi deux objectifs contradictoires : la maximisation d'un paramètre conduit à la minimisation de l'autre.

4.4.2 Optimisation multi-objectifs

Un problème d'optimisation est dit multi-objectifs lorsque plusieurs objectifs doivent être minimisés en même temps. Nous souhaitons trouver un front de solutions non-dominée de Pareto, qui est l'ensemble des valeurs qui sont à l'optimum de Pareto. Une solution est à l'optimum de Pareto quand il est impossible d'améliorer un objectif sans réduire au moins un autre. La méthode la plus courante pour résoudre un problème multi-objectifs consiste à affecter des poids sur chaque fonction objective que l'on cherche à minimiser puis de les sommer afin de se ramener a un problème avec un seul objectif. Cependant cette approche nécessite de recalculer explicitement à chaque fois que l'on souhaite un compromis différent entre les différents objectifs. Or nous voulons déployer différentes configurations $\mathscr{C}$ au niveau du RPCA et le laisser choisir la configuration la plus pertinente en fonction des conditions réseau (Reconfiguration topologique, pic de trafic, ...).

Nous utilisons une méthode d'optimisation multi-objectif appelée Non-dominated Sorting Genetic Algorithm (NSGA) II [29] reposant sur des algorithmes génétiques. L'approche que nous avons choisie se justifie aussi par le fait qu'un algorithme génétique génère une population de solutions. Ainsi l'algorithme donne plusieurs solutions à un même problème qui peuvent être stockée sur le RPCA et déployées dynamiquement en fonction des conditions réseaux sans nécessiter des relancer un programme d'optimisation. Il est à noter que ce processus d'optimisation peut être déportée sur un serveur spécialisé et n'est pas obligatoirement hébergé sur le RPCA. En outre, les algorithmes génétiques offrent une approche plus versatile ne nécessitant aucune hypothèse de convexité sur les fonctions étudiées.

Les algorithmes génétiques peuvent trouver plusieurs optimums locaux et ne pas rester bloqués sur un seul comme d'autres méthodes à base de gradient [78]. Les fonctions objectives peuvent être quelconques avec un grand nombre de paramètres comme c'est le cas avec notre modélisation de la durée de vie du LLN. De plus à l'inverse des méthodes d'optimisations complexes qui supposent une connaissance complète du problème, les algorithmes génétiques peuvent être utilisé avec aucune connaissance des fonctions objectives (black box).

Cependant les algorithmes génétiques ont des inconvénients. Dans le cas où des méthodes convexes sont disponibles, elles permettent de converger plus rapidement et de manière déterministe vers un optimum. Les méthodes génétiques ne fournissent pas de garanties que la population de solutions converge rapidement vers l'optimum. En outre, rien ne prouve qu'une solution donnée va s'améliorer d'une génération à l'autre. Ainsi si c'est là rapidité ou le déterminisme qui est privilégiée et qu'une modélisation du problème sous forme de fonctions convexes est disponible alors les méthodes convexes peuvent être une alternative à l'utilisation d'algorithmes génétiques.

Les algorithmes génétiques sont connus et utilisés dans les LLNs fils pour avoir un placement optimal des nœuds [41, 58]. Cependant l'utilisation de méthodes génétiques pour trouver les paramètres optimaux d'un RPC n'ont pas encore été explorés.

4.4.3 Formalisation en algorithme génétique

Un algorithme génétique utilise une population individus de base qui est représenté dans l'étape 1 sur le figure 4.4. Chaque individus corresponds à une solution possible à problème donné. Dans notre cas, les individus seront des $\mathscr{C}$ correspondant au temps de vie en cache pour chaque URI que le RPC gère. La population initiale est générée de manière aléatoire entre c_{min} et c_{max}. À chaque itération (appelée génération) l'algorithme génétique va croiser des individus, affecter des mutations et sélectionner un certain nombre d'individus pour recommencer à la génération suivante. L'algorithme s'exécute pendant un nombre prédéfini de générations. Les meilleures solutions qui ont été sélectionnées au fil des générations sont données comme solution aux problème et constitue le front de Pareto après un nombre suffisant d'étapes.

4.4.3.1 Aptitude (Fitness)

La fonction d'aptitude évalue a quel point une solution est admissible ou non et si elle l'est quelle est son efficacité. Elle est utilisée comme base pour sélectionner des individus d'une génération à l'autre. Déterminer une fonction d'aptitude est la partie la plus liée au problème multi-objectifs lors de la conception d'un algorithme génétique ; les autres étapes étant relativement génériques d'un problème à l'autre.

Nous définissons ainsi f la fonction d'aptitude en normalisant les grandeurs $\mathscr{S}(\mathscr{C})$ et $\mathscr{L}(\mathscr{C})$ entre 0 et 1. Nous prenons la moyenne afin de ne pas privilégier des solutions conservatrices en énergie au détriment d'autres.

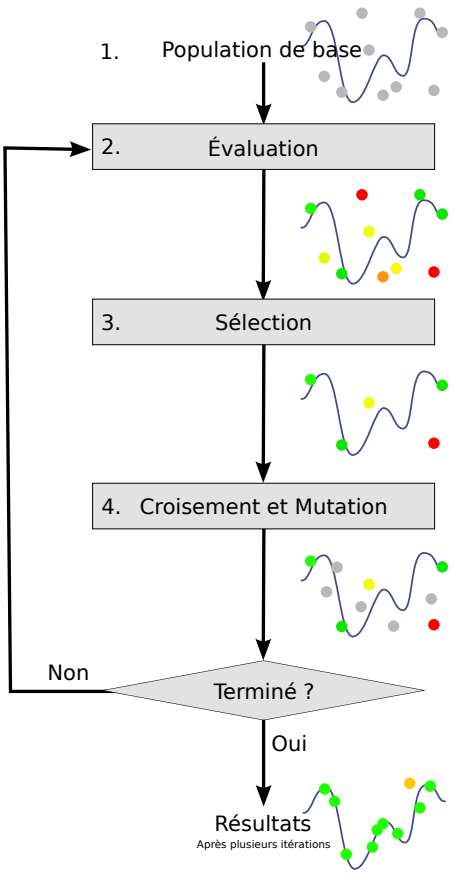


FIGURE 4.4 – Schéma des étapes d'un algorithme génétique

$$f(\mathscr{C}) = \frac{1}{2}\left(\frac{\mathscr{S}(\mathscr{C})-\mathscr{S}(\mathscr{C}_{\min})}{\mathscr{S}(\mathscr{C}_{\max})-\mathscr{S}(\mathscr{C}_{\min})} + \frac{\mathscr{L}(\mathscr{C})-\mathscr{L}(\mathscr{C}_{\min})}{\mathscr{L}(\mathscr{C}_{\max})-\mathscr{L}(\mathscr{C}_{\min})}\right) \tag{4.6}$$

$$f(\mathscr{C}) = \frac{1}{2}\left(\mathscr{S}(\mathscr{C}) + \frac{\mathscr{L}(\mathscr{C})-\mathscr{L}(\mathscr{C}_{\min})}{\mathscr{L}(\mathscr{C}_{\max})-\mathscr{L}(\mathscr{C}_{\min})}\right) \tag{4.7}$$

Puisque nous avons $\mathscr{S}(\mathscr{C}_{\min}) = 0$ et $\mathscr{S}(\mathscr{C}_{\max}) = 1$, nous pouvons simplifier l'équation 4.6 pour obtenir 4.7.

4.4.3.2 Sélection

Le processus de sélection choisit quels individus survivent d'une génération à l'autre. Un des écueils courant des méthodes d'optimisation multi-objectifs génétique est de rester bloqué dans un sous ensemble de solutions. A niveau d'aptitudes semblables, les solutions les plus éloignées les unes des autres devraient être sélectionnées. C'est la méthode utilisée par NSGA-II [30] qui est élitiste : les meilleures solutions sont retenues d'une génération à l'autre et forment les élites. Cependant cette sélection s'accompagne d'une "prime à la diversité" ce qui privilégie les solutions éloignées les unes des autres.

4.4.3.3 Croisement & Mutation

La mutation est un processus de transformation qui va être utilisé sur un individu avec une probabilité p_m sur chaque à chaque génération. Nous avons utilisé le processus de mutation polynomial qui est celui utilisé dans NSGAII et détaillé dans [30]. Ce processus produit une solution mutante pour chaque gène (en l'occurrence un c_u) est dans les bornes inférieures et supérieures admissibles (en l'occurrence $c_{\min}$ et $c_{\max}$). La mutation polynomiale (aussi appelée convolution gaussienne) [78] consiste à appliquer un bruit gaussien (μ, σ) sur chaque composant de notre solution. Nous avons utilisé comme borne de la distribution c_{min} et c_{max}.

Le croisement est un mécanisme d'exploration des solutions possibles. C'est un opérateur d'algorithme génétique utilisé pour mélanger les gènes de deux solutions pour en former deux autres. En l'occurrence deux sous ensemble de c_u, c'_u appartenant à deux solutions distinctes $\mathscr{C}$ et $\mathscr{C}'$ seront échangés à la génération suivante si un croisement se produit pour ces deux solutions. De nombreux mécanismes de croisements sont disponibles (point unique de croisement, point double, ...) [78] nous avons utilisé un mécanisme à point double représenté sur la figure 4.5 car il permet de mitiger les problèmes de sélection des premiers composants des vecteurs. En effet avec un croisement à point simple les premiers indices des vecteurs sont le plus souvent sélectionnés ce qui induit un biais.

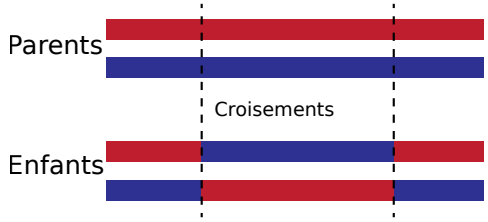


FIGURE 4.5 – Croisement de deux individus par points doubles

4.4.4 Validation expérimentale du RPCA

Le choix de valeur adéquate pour c_u introduit un compromis entre la durée de vie du réseau et la satisfaction des utilisateurs. En outre, comme certains nœuds agissent comme routeurs, ils influent sur le nombre de transmissions qu'ils devront gérer.

Nous avons utilisé la même topologie radio utilisée dans les expériences précédentes et représentée sur la Figure 4.1. Nous avons mis les contraintes sur la durée de vie du réseau à respectivement 25, 50, et 75 jours, et nous avons évalué l'ensemble c^* qui maximise la satisfaction des utilisateurs tout en remplissant la durée de vie du réseau. A titre de comparaison nous montrons les résultats obtenus avec $c_i = c_{min}$ et $c_i = c_{max}$. Les résultats sont montrés dans la table **??**.

4.4.4.1 Compromis entre durée de vie et satisfaction utilisateur

Pour simplifier, seul les modules de communications seront inclus dans notre modèle énergétique détaillé dans l'annexe A. Nous supposons que tous les nœuds possèdent la même énergie initiale. Comme présenté dans la Figure 4.6, nous obtenons pour chaque durée de vie admissible la meilleure satisfaction utilisateur possible. Nous obtenons ainsi un front de Pareto permettant de choisir parmi les solutions optimales celle qui est adaptée à notre application en fonction des informations provenant des différentes couches logiques sans avoir à recalculer une solution au problème multi-objectif.

La raison pour laquelle nous obtenons un front de Pareto en forme de droite provient de nos hypothèses de départ. En effet, le rythme d'arrivée des requêtes est supposé uniforme, et tous les

Population	P	100
Probabilité d'accouplement	p_c	0.5
Probabilité de mutation	p_m	0.2
Nombre de génération	n_g	50
Index de la distribution de mutation	η_m	20
Temps de validité en cache minimal	c_{min}	1
Temps de validité en cache maximal	c_{max}	9
Taux de requêtes entrantes	λ	1

TABLE 4.2 – Paramètres utilisés dans l'optimisation multi-objectifs.

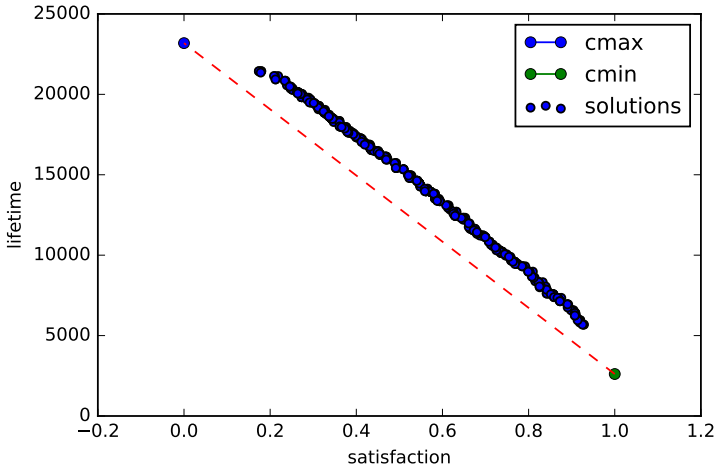


FIGURE 4.6 – Front de Pareto pour le scénario envisagé.

nœuds sont identiques. Dans ces conditions, nous obtenons une loi linéaire sur les défauts de cache et la consommation du réseau qui en dépend. Cependant, dans le cas où la couche MAC pourrait être dynamiquement configurée en fonction des informations disponibles, la relation ne serait plus linéaire.

Selon les résultats précédents, il est possible de trouver un ensemble de solutions appropriées qui permettent d'obtenir la meilleure configuration des durées de vie de cache. Le solveur associés à chacun de ces points un ensemble de paramètres c_i qui peuvent être utilisés sur le réseau start-up à régler correctement le la durée de cache.

La fonction économique que l'on va utiliser peut être modifié selon différents critères. Par exemple on peut avoir une répartition des poids différentes selon qu'un nœud a plus ou moins de voisins. Une autre approche consiste à prendre en compte la répartition des popularités des requêtes envoyées.

Il y a de très nombreux paramètres (les durées de vie de cache, les bornes la satisfaction qui peut prendre des formes très diverses et non-linéaires) Fixer une contrainte de durée de vie ne donne pas trivialement une solution de configuration de cache. Si on a une fourche par exemple on peut avoir plusieurs solutions qui aboutissent a la même durée de vie sur le réseau. D'où l'intérêt d'avoir une variété de solutions larges afin d'offrir de nombreuses opportunités.

4.4.4.2 Répartition des temps de vie pour les nœuds

le calcul de la durée de vie retourne l'index du nœud qui épuise sa batterie en premier.

4.4.4.3 Répartition des satisfactions pour les nœuds

Le calcul de la satisfaction est immédiat pour une durée de cache donnée. Ainsi,

4.5 Conclusion

Ce chapitre a présenté une stratégie pour améliorer la qualité de service d'un réseau de capteurs en adaptant dynamiquement les paramètres d'un RPC en fonction d'informations en provenance de différentes couches logiques du système. En premier lieu, nous avons vu qu'un RPC standard aidait à économiser de l'énergie puisqu'il empêche les communications redondantes triviales. Puis nous avons introduit un modèle d'optimisation exploitant les différents critères des utilisateurs ainsi que les informations extraites du cache. Ce qui nous a permis de configurer de manière optimale la gestion de la durée de vie des informations au sein du RPCA.

Ce chapitre a abordé le problème de l'énergie Quality of Service (QoS) efficaces optimisation utilisant des techniques entre couches et exploitant une spécifiquement introduites plate-forme de mise en cache. Nous avons d'abord présenté la mise en œuvre d'une mise en cache solution basée sur un nœud proxy qui est en charge de répondre, si un cache valeur est disponible, à une demande provenant d'un client distant sans transférer au LLN. Les résultats de simulation montrent que l'introduction d'un l'architecture de mise en cache a un impact positif en termes d'économie d'énergie sur le système le rendement, car il permet de réduire les transmissions à l'intérieur du LLN. Alors, nous avons introduit une méthode d'optimisation qui, exploitant les informations recueillies par le RPL protocole et donné un ensemble de contraintes sur le minimum et les valeurs maximales de la durée de cache, permet de configurer de manière optimale les valeurs des durées de vie de mise en cache. La stratégie d'optimisation proposée permet soit trouver des solutions adaptées à la présence de contraintes sur la durée de vie du réseau ou à savoir l'ensemble non-dominée optimale de solutions dans le cas d'optimisation multi-objectifs.

Une autre ouverture serait d'étendre à un nombre quelconque de ressources concurrentes, chacune ayant des popularités et une satisfaction sur chaque ressource différente. Dans ce cas là une piste de recherche serait une modélisation s'inspirant de Knapsack Problem (KP).

Publications

— Rémy Leone, Paolo Medagliani et Jérémie Leguay. Optimizing QoS in Wireless Sensors Networks using a Caching Platform. *Sensornets 2013*, page 56, Barcelone, Espagne, Février 2013.

— Rémy Leone, Paolo Medagliani et Jérémie Leguay. Optimisation de la qualité de service par l'utilisation de mémoire cache 15èmes Rencontres Francophones sur les Aspects Algorithmiques des Télécommunications

Chapitre 5

Mesure implicite de la consommation énergétique d'un LLN

> There are more things in heaven and earth, Horatio, than are dreamt of in your philosophy
>
> Shakespeare - Hamlet (1.5.167-8)

Contents

Dans ce chapitre, nous présentons un estimateur de trafic et de consommation énergétique utilisant la topologie de routage et le trafic observé à la passerelle pour prédire de la consommation énergétique des nœuds. Grâce à des simulations, nous comparons nos estimations par rapport aux valeurs

de consommation énergétique réelle recueillie par le simulateur. Nous montrons que la connaissance de la topologie dans le cas d'un réseau maillé multi-saut améliore la précision de la supervision passive et que l'impact des protocoles applicatifs est facilement prévisible. De plus, nous montrons que le biais de la supervision passive (non détection des multiples tentatives de transmissions, des collisions de paquets et de la contention) peuvent être découvert et corrigé par une supervision active.

Nous introduirons dans la section 5.1 les concepts principaux utilisés dans le chapitre et la justification générale de notre approche en comparaison avec l'état de l'art ainsi que notre contribution. Les modèles seront exposés dans la section 5.2. Puis nous développerons notre approche de supervision passive dans la section **??** et testerons sa précision, ses limites et comment les corriger. Finalement la section 5.4 terminera ce chapitre et proposera des ouvertures à ces travaux.

5.1 Introduction

La passerelle peut être vue comme la source et la destination d'une grande partie du trafic applicatif pour le LLN. Elle joue un rôle de référence dans plusieurs protocoles comme RPL [134] ou bien des couches MAC comme IEEE 802.15.4e [91]. Plus récemment, la passerelle peut même orchestrer l'accès au médium comme dans le cas de 6TiSCH [1]. La passerelle est dans de nombreux déploiement le nœud qui a la connaissance la plus détaillée du réseau car elle relaie le trafic applicatif entrant et sortant du LLN. Ainsi elle peut acquérir une bonne vue du réseau en terme de (i) topologie, (ii) ressources offertes, (iii) allocation des ressources radio, et (iv) l'état de fonctionnement des nœuds. Avec cet ensemble de connaissance, la passerelle peut fournir des services de supervision du réseau. Un exemple de supervision est une estimation de la consommation énergétique et indirectement la durée de vie des nœuds.

L'architecture que nous proposons est exposée dans la figure 2.4. Le mécanisme de supervision passive observe le trafic réseau rentrant et sortant du LLN ainsi il produit une trace de l'activité réseau qui sera envoyé à l'estimateur. Si un protocole de routage permettant à la racine de connaitre toute la topologie sous-jacente du réseau est disponible alors cette topologie est également envoyé à l'estimateur. L'estimateur produit une estimation de la consommation énergétique pour l'ensemble des nœuds.

Cependant ce mécanisme de supervision est biaisé, il ne peut pas prendre en compte les phénomènes locaux qui ne passent pas par la passerelle ce qui conduit à des prévisions sous estimées. Quand cela est possible, un mécanisme de supervision active peut être introduit pour corriger le biais de la supervision passive.

5.1.1 Motivations

Superviser un LLN est essentiel pour s'assurer qu'il est dimensionné, provisionné et fonctionne de manière nominale. La supervision vise à apporter à l'administrateur du LLN toutes les informations nécessaires afin de prendre de bonnes décisions sur son provisionnement. L'administrateur n'a cependant pas toujours connaissance du code s'exécutant sur un nœud ce qui limite sa vision. De plus même quand des fonctionnalités de supervision actives sont disponibles, il peut être couteux de les utiliser puisque les ressources en bande passante sont très limitées. Ainsi des méthodes passives pour mesurer l'état d'un nœud et en particulier sa consommation énergétique peuvent être pertinentes et proposer une approche complémentaire à la supervision active.

Superviser un LLN est essentiel pour s'assurer qu'il est dimensionné, provisionné et fonctionne de manière nominale [74]. La supervision peut déclencher des reconfigurations des noeuds lorsque les situations s'y prêtent ou prévoir quand leur réserve d'énergie s'épuisera. Afin de prédire de manière

efficace la durée de vie du LLN, il est nécessaire de avoir une vue sur la consommation énergétique. Toutefois, transmettre des informations de supervision a un cout et dans des déploiement où la bande-passante est rare et qu'un grand nombre de noeuds sont déployés, une politique de supervision périodique naive [1] peut introduire des couts importants. De plus certains noeuds peuvent ne pas offrir des fonctionnalités d'introspection rendant leur supervision difficile. Effectuer des mesures de consommation énergétique indirectes et passives peut être une alternative aux politiques de supervision classiques. Elles permettent de fournir une estimation de l'état des noeuds tout en consommant aussi peu d'énergie que possible pour l'obtenir.

5.1.2 État de l'art

Quelques contributions ont examiné le problème général de la surveillance d'un LLN efficace en énergie. Par exemple, [75] et [68] considèrent le problème de la sélection d'un sous-ensemble de capteurs "sondeurs" chargés de surveiller activement les autres capteurs "sondés". Les sondeurs sont en charge d'émettre des alarmes vers la passerelle si ils détectent une anomalie. [75] propose un algorithme distribué d'approximation pour sélectionner un nombre minimum de sondeurs et étudie le taux de faux positif généré. [68] propose de réduire la dépense énergétique en utilisant des paquets de contrôle de routage pour sélectionner les sondeurs et en intégrant les rapports de suivi dans les messages de contrôle du protocole de routage. Ces approches nécessitent des sondes déployées dans le réseau tandis que nous attendons que la supervision active soit facultative.

Dans [19], les auteurs proposent LiveNet, une architecture de surveillance semi-passive qui repose sur les sondes situés dans le réseau. En utilisant les traces agrégées transmises à la passerelle, LiveNet est capable de reconstruire topologie de réseau et de déterminer divers paramètres de performance du réseau. Ce travail vise explicitement surveillance de l'énergie, mais pourrait être adaptée à d'autres indicateurs de performance. Cependant, il nécessite la transmission et le traitement de traces dans le réseau, il est donc pas entièrement pertinent pour notre objectif qui se place exclusivement au niveau de la passerelle.

Dans [140], les auteurs introduisent une méthode distribué pour créer un carte de l'énergie restante d'un LLN [2]. Les nœuds déclarent leur niveau d'énergie résiduelle à un voisin nœud, en charge de l'agrégation et la compression de ces informations et ne transmettent que des mises à jour incrémentielles (condensés) à la passerelle. Suivant cette idée, [84] laisse l'estimation et la prédiction de temps de vie à chaque nœud puis se charge de l'envoyer au moniteur de réseau. De plus, [84] compare une méthode probabiliste, basée sur les chaînes de Markov, et une méthode statistique, sur la base d'un modèle auto-régressif, avec un simple, méthode de déclaration explicite. Dans [56], les auteurs étendent cette idée en modélisant l'énergie de chaque nœud avec un modèle de Markov caché dont les coefficients sont l'écoute avec des mesures explicites. Dans [18], les auteurs construisent une carte de l'énergie et changent la structure de surveillance régulièrement pour redistribuer le coût de cette surveillance de façon équitable à travers le réseau. Si l'idée de construire une carte de l'énergie du réseau est étroitement liée à notre premier but, toutes les méthodes mentionnées ci-dessus reposent fortement sur les rapports de l'énergie explicite et continus des nœuds alors que notre approche est passive.

5.1.3 Contribution

Nous proposons d'étudier la précision des mécanismes de supervision passive qui peuvent être déployés au niveau de la passerelle. La supervision passive dans notre étude visera à prévoir la consom-

1. Envois synchronisés causant des congestions ou retransmissions, intervalle périodique trop grands ou trop petits
2. Aussi appellée weathermap dans l'industrie

mation énergétique des noeuds en utilisant le trafic de la couche réseau intercepté à la passerelle comme indice pour inférer la consommation de la radio de chacun des noeuds.

En observant le trafic réseau passant à la passerelle à destination d'un noeud nous pouvons créer un modèle du trafic réseau et donc de la consommation des noeuds impliqués. Dans le cas des topologies multi-sauts, nous prenons également en compte les consommations engendrées par le relayage des paquets. Ces informations sur la topologie sont combinées avec une modélisation des protocoles pour estimer le temps passé par un nœud à transmettre et recevoir des messages de la part de ses voisins.

Puisque la supervision fonctionne depuis la passerelle, elle ne saisi pas la consommation énergétique induite par les comportements radio des noeuds à un niveau local imprévisibles par la passerelle. Pour cette raison, nous introduisons des mecanismes de supervision active qui peuvent corriger nos estimations et corriger nos biais d'observations.

- Diagnostics de problèmes - Prévision des pannes causées par des chutes d'énergies

5.2 Modélisation de la consommation énergétique d'un LLN

Dans la plupart des LLN, l'interface radio est la principale consommatrice d'énergie [4]. Le micro controleur et le reste du système opère à des fréquences basses et seul les opérations d'écriture de la mémoire flash ont besoin d'une énergie comparable à celle utilisée par la radio [85]. En particulier, l'énergie consommée par la transmission et la réception sont en générale similaire [31]. Une grande utilisation de la radio impliquera une grande quantité d'énergie consommée. Dans le reste du chapitre, nous ne considérons que la consommation de l'interface radio.

En IEEE 802.15.4, quand une source émet une trame, tous les nœuds éveillés qui sont à portée radio vont tenter de décoder la trame et devront la saisir complètement avant de vérifier qu'elle est correcte par sa somme de contrôle et qu'elle est destiné au nœud en question. Il est possible qu'un voisin éteigne sa radio lorsqu'il détecte qu'il n'est pas concerné par cette transmission. Néanmoins, chaque nœud réveillé dépense toujours une quantité minimale d'énergie, pour analyser une trame qui est émise par l'un de ses voisins [3].

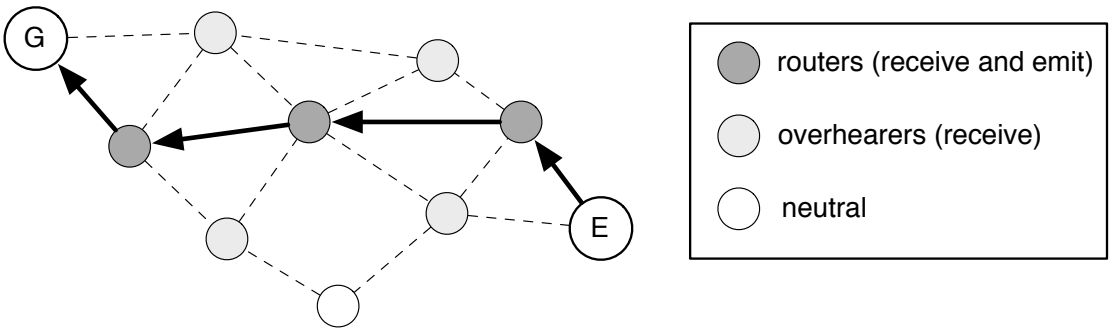


FIGURE 5.1 – Nœuds impactés par l'acheminement d'une trame depuis le nœud E vers le nœud G.

Si nous regardons un nœud i et $\mathcal{N}(i)$, ses voisins qui sont à portée de transmission. i transmet une trame f de $\mathcal{L}(f)$ octets à un autre nœud $j \in \mathcal{N}(i)$ tous les nœuds non-endormis dans $\mathcal{N}(i)$ consomment de l'énergie pour écouter la trame. Nous considérons que les nœuds qui appartiennent $\mathcal{N}(i) \setminus \{j\}$ et ceux que ne sont pas endormis peuvent limiter la réception et le traitement des trames

3. Une couche MAC efficace peut mitiger cet effet en gérant les réveils et endormissement [47, 1].

qui ne leur sont pas destiné d'une taille $\mathscr{U}(f)$ octets avant qu'elle n'éteigne leur interface radio. L'énergie dépensée pour la transmission de cette trame est :

$$\begin{cases} \text{Emitter (node } i) & : S_{cost}(\mathscr{L}(f)) \\ \text{Receiver (node } j) & : R_{cost}(\mathscr{L}(f)) \\ \text{Awake over-hearers}(\mathscr{N}(i) \setminus \{j\}) & : R_{cost}(\mathscr{U}(f)) \end{cases}$$

$$E_i(t) = \sum_{j \in \mathscr{N}(i)} \left(\sum_{f \in \mathscr{F}_{i,j}(t)} \underbrace{S_{cost}(\mathscr{L}(f))}_{\text{Emissions}} + \sum_{f \in \mathscr{F}_{j,i}(t)} \underbrace{R_{cost}(\mathscr{L}(f))}_{\text{Receptions}} + \sum_{k \in \mathscr{N}(j) \setminus \{i\}} \sum_{f \in \mathscr{F}_{j,k}(t)} \gamma . \underbrace{R_{cost}(\mathscr{O}(f))}_{\text{Overhearing}} \right),$$

où $\gamma \in [0;1]$ modélise la fraction de trames qui seront entendues durant le cycle de veille. En utilisant les mêmes notations, nous pouvons aussi exprimer le coût d'envoi d'une trame depuis un nœud i vers un nœud j pour l'émetteur et le récepteur. L'accès au canal dans IEEE 802.15.4 rends $\mathscr{L}(f)$ comme une fonction linéaire de la taille de trame et dans la plupart des implémentations [34] $\mathscr{U}(f) = \mathscr{L}(f)$.

Puisque pour une plateforme fixée, la consommation énergétique de ces états est connue, nous pouvons calculer l'énergie dépensée par un nœud en sachant le temps qu'il a passé dans chaque états.

5.2.1 Consommation énergétique de transmission et réception

Nous devons calculer combien d'énergie est dépensé par chaque nœud pour transmettre une trame f de taille $\mathscr{L}(f)$ octets envoyés en unicast et avec un acquittement.

IEEE 802.15.4 fournit le temps requis pour transmettre la trame f de la manière suivante :

$$T_p(f) = \left(\frac{8\mathscr{L}(f)}{R} + \left\lceil \frac{\mathscr{L}(f)}{L} \right\rceil h \right)$$

où $\mathscr{L}(f)$ est la taille d'une trame IEEE 802.15.4 (exprimée en octets), $R = 250$ kbit/s est le débit du IEEE 802.15.4, L est la charge utile (payload) maximale d'une trame IEEE 802.15.4 (127 octets) et h est le temps requis pour transmettre l'entête d'une trame. Le standard IEEE 802.15.4 fournit $h = 992\mu s$ pour les entêtes ce qui est confirmé en simulation. Puisque les entêtes sont envoyées pour toutes les trames nous prenons aussi en compte la surcharge causée dans le cas d'une fragmentation de paquets.

Soit $P_{\mathscr{T}}$ et $P_{\mathscr{R}}$ les puissance requises pour emetre et recevoir des données respectivement. Si on multiplie $T_p(f)$ par $P_{\mathscr{T}}$ (respectivement $P_{\mathscr{R}}$) nous obtenons l'énergie depensée pour transmettre (respectivement recevoir) f. Les paramètres peuvent être trouvés dans les notices d'utilisations des composants utilisés. Par exemple, la radio Chipcon CC2420 [20] implémentant IEEE 802.15.4 opère avec une tension de $V_{\text{DD}} = 3V$, le courant pendant la réception est de $I_{\mathscr{R}} = 19.7\,mA$ et le courant durant une émission à 0 dBm est $I_{\mathscr{T}} = 17.4\,mA$ [27].

Nous pouvons donc estimer $S_{\text{cost}}(f)$ et $R_{\text{cost}}(f)$, l'énergie nécessaire pour respectivement envoyer et recevoir une trame f :

$$S_{\text{cost}}(f) = P_{\mathscr{T}} N_{\text{sender}}(f) T_p(f) + P_{\mathscr{R}}\, t(ACK) \tag{5.1}$$

$$R_{\text{cost}}(f) = P_{\mathscr{R}} N_{\text{receiver}}(f) T_p(f) + P_{\mathscr{T}}\, t(ACK) \tag{5.2}$$

où $N_{\text{sender}}(f)$ et $N_{\text{receiver}}(f)$ sont le nombre de tentatives entreprises par l'expéditeur et le destinataire respectivement.

Dans le reste du chapitre nous utiliserons $S_{cost}(f)$ pour désigner l'énergie pour envoyer une trame de $\mathscr{L}(f)$ octets. Cette énergie prends en compe la procédure d'accès au canal et le préambule de transmission si il y en a un. De même, $R_{cost}(f)$ désigne l'énergie dépensée par un nœud pour recevoir une trame de $\mathscr{L}(f)$ octets.

Dans le reste du scénario, nous considérons que le trafic réseau est suffisamment faible pour que les collisions soient négligéables. Cependant, en raison de leurs large périodes d'endormissement, une désynchronisation des nœuds est possible et un emetteur et un récepteur ne seront pas forcément réveillé au même instant ce qui peut engendrer du strobbing dans le cas de ContikiMAC que nous devons prendre en compte.

5.2.2 Strobbing de ContikiMAC

Lorsque ContikiMAC est utilisé, un expéditeur doit transmettre plusieurs fois une trame avant de recevoir un acquittement. Même avec des mécanismes de verrouillage de phase de ContikiMAC, les horloges dérivent et plusieurs tentatives peuvent être nécessaire pour l'envoi d'une trame. Ce cycle de veille repose sur IEEE 802.15.4 comme nous l'avons vu dans 2.1.2 et utilise Carrier Sense Multiple Access with Collision Avoidance (CSMA/CA) pour éviter les collisions et écoute toujours quand la radio ne transmet pas afin de garantir une certaine stabilité du réseau. Ces mécanismes ajoutent des couts à la transmission théorique d'une trame. Pour mesurer ces couts, nous effectuons une simulation avec une source et une destination qui échangent du trafic. Nous mesurons combien d'essais sont nécessaire en moyenne avec seulement deux nœuds. Dans cette expérience puisqu'il n'y a qu'un expéditeur et un seul destinataire, on évite les collisions de paquets d'un tiers. Nous comptons combien de fois un paquet est envoyé par l'expéditeur avant d'être acquitté par le destinataire.

Nous introduisons l'expérience suivante : deux noeuds ContikiMAC communiquent l'un avec l'autre avec un trafic constant. Nous mesurons le nombre moyen de transmissions nécessaires pour qu'une trame soit acquittée par le destinataire avec 10 essais pour chaque point.

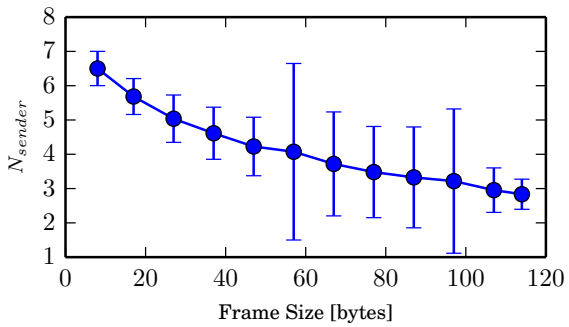


FIGURE 5.2 – Nombre moyen de tentatives d'envois en fonction de la taille de la trame.

La figure 5.2 représente la moyenne du nombre de transmission $N_{\text{sender}}(f)$ nécessaire à l'envoi d'une trame de $\mathscr{L}(f)$ octets entre deux nœuds. Le nombre d'envoi moyen nécessaire pour envoyer un long paquet est plus faible que pour en envoyer un court. C'était à prévoir sachant que les trames longues mettent plus de temps pour être transmises et ont donc plus de chance d'être écouté pendant un réveil du destinataire. Comme nous l'avons vu dans la section **??**, ContikiMAC induit le fait d'envoyer plusieurs fois un paquets avant qu'il ne soit acquitté. Ces multiples tentatives ne peuvent pas être prédite facilement pour un nœud à un instant précis cependant des comportement moyens peuvent être connus pour une taille de paquet donnée. En outre, nous pouvons dire que c'est l'expéditeur qui va dépenser le plus d'énergie dans cette configuration. Comme les paquets ACK ont une taille constante, le cout de réception pour le destinataire est constant.

De son côté, le destinataire se réveille en moyenne au milieu d'une transmission et attends le prochain essai d'envoi pour recevoir la trame complètement et envoyé son acquittement. Le nombre de trames reçu peut être estimé à $N_{\text{receiver}} = 1.5$. La passerelle n'ayant pas de cycles de veille, pour les nœuds en contact direct avec elle on a $N_{\text{sender}}(f) = 1$.

5.2.3 Supervision passive sans connaissance de la topologie

La supervision passive sans connaissance de la topologie n'estime l'impact que sur la source et la destination des paquets. Dans le scénario présenté sur la figure. 5.1 dans lequel un nœud E envoit une trame à la passerelle G, l'estimation naïve ne déduit que l'énergie consommée par les noeuds E et G. Tous les autres nœuds appartenant au réseau sont ignorés.

Soit $\mathscr{D}_i$ les paquets provenant de i et $\mathscr{A}_i$ les paquets autant pour destination finale i. Nous obtenons l'énergie estimée suivante :

$$S_i(t) = \sum_{f \in \mathscr{D}_i(t)} S_{cost}(f)$$

$$R_i(t) = \sum_{f \in \mathscr{A}_i(t)} R_{cost}(f)$$

$$\widehat{E}_i(t) = \sum_{m \in \mathscr{D}_i(t)} S_{\text{cost}}(m) + \sum_{m \in \mathscr{A}_i(t)} R_{\text{cost}}(m).$$

Ce type de supervision passive est adapté pour des topologies en étoile à un saut et peut être déployé sans engendrer de trafic de supervision supplémentaire. Cependant il ignore les couts engendrés dans des scénarios multi-sauts par les retransmissions de paquets.

5.2.4 Supervision passive avec connaissance de la topologie

Un nœud peut être le destinataire d'un paquet réseau mais peut aussi être utilisé comme relais dans un chemin multi-sauts. La topologie devient donc un paramètre déterminant pour savoir si un nœud passera beaucoup de temps à relayer des paquets pour ses voisins et donc utilisera une grande partie de ses ressources à cette fin. Ainsi lorsque les informations sur le routage sont disponibles il est possible de compléter la supervision passive pour prendre en compte l'impact énergétique causé par le relayage des paquets comme montré en gris sombre sur la figure 5.1).

Soit $\mathscr{F}_i$ l'ensemble des trames qui sont retransmises par le nœud i alors qu'il n'est ni la source ni le destinataire du paquet.

$$\widehat{E}_i(t) = \sum_{f \in \mathscr{D}_i(t) \cup \mathscr{F}_i(t)} S_{\text{cost}}(f) + \sum_{f \in \mathscr{A}_i(t) \cup \mathscr{F}_i(t)} R_{\text{cost}}(f)$$

5.2.5 Supervision active pour la correction des biais

La précision de la supervision passive peut être améliorée en la combinant avec des messages de supervision active. Comme nous l'avons vu dans les parties précédentes, la supervision passive permet de prévoir une partie de la consommation énergétique des nœuds. Lorsqu'elle est disponible, la supervision active permet d'avoir des relevés plus précis. Nous démontrons comment la passerelle peut apprendre les biais induits de la supervision passive. Le but étant de réduire la quantité de supervision active nécessaire pour avoir une bonne vue du réseau.

Nous considérons des messages de supervisions explicites envoyés par le nœud vers la passerelle. Les messages contiennent les temps mesurés par le nœud passé dans chaque état de transmission.

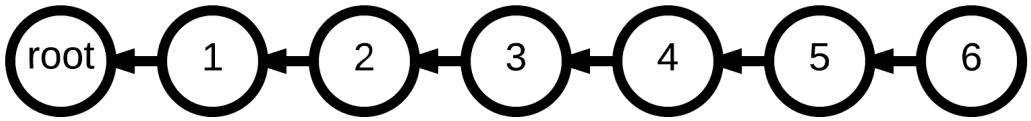


FIGURE 5.3 – Topologie réseau

Notre objectif est de trouver quel est le rythme optimal d'envoi en fonction des conditions réseau. Quand la passerelle reçoit un message de supervision explicite au temps t elle recalcule son estimateur de consommation énergétique pour le nœud $\widehat{E}(t)$ de la manière suivante :

$$\widehat{E}(t) = E(t_{\mathrm{r}}) + R_i(t) + S_i(t) + \epsilon(t_{\mathrm{r}})\frac{(t - t_{\mathrm{r}})}{T} \tag{5.3}$$

$$\epsilon(t_r) = \alpha.(E(t_{\mathrm{r}}) - \widehat{E}(t_{\mathrm{r}})) + (1-\alpha).\epsilon(t_{\mathrm{r}-1}) \tag{5.4}$$

où t_{r} et $t_{\mathrm{r}-1}$ sont les deux dernières dates de message de supervision active. S_i et R_i sont respectivement les couts estimés d'envoi et de réception induis par le trafic applicatif depuis le dernière supervision active et $E(t_r)$ est le niveau d'énergie consommé réel du nœud à t_r. $\epsilon(t_{\mathrm{r}})$ est l'estimation de l'erreur apprise des précédents messages de supervision active. Il doit intégrer les consommation énergétique manquée par notre estimateur de base comme le trafic non routés vers la passerelle ou bien les pertes de paquets.

$\frac{E(t_r)-E(t_{r-1})}{t_r - t_{r-1}}$ représente la tendance de l'énergie consommée et prends en compte les paquet émis par le réseau y compris ceux que la passerelle n'a pas pu prévoir. Notons que $\epsilon(0) = 0$.

5.3 Validation expérimentale

5.3.1 Résultats expérimentaux - Topologie en chaine

Nous prouvons expérimentalement que la connaissance des routes améliore les estimations. Nous avons évalué la précision des estimations en utilisant le simulateur COOJA avec son extension Powertracker qui mesure le temps que chaque nœud émulé passe dans un état de consommation énergétique précis avec une résolution de $1\mu s$. Les nœuds utilisent Contiki comme système d'exploitation, RPL [134] comme protocole de routage et ContikiMAC [34] sur IEEE 802.15.4.

Considérons une topologie simple à 7 nœuds comme réprésenté sur la figure 5.3. Les nœuds ne peuvent envoyer et recevoir de paquets que de la part de leurs voisins adjacents. Dans ce scénario durant 200 secondes, chaque nœud envoit à la racine un paquet User Datagram Protocol (UDP) avec une charge utile de 10 octets (trame de 69 octets) chaque seconde. Le trafic "applicatif" démarre lorsque la topologie réseau est établie. Nous utilisons une valeur moyenne de strobbing de 3.76 comme trouvé dans les expériences précédentes de calibration dans la section 5.2.2. Nous utiliserons les données présentées dans la Table 5.1 pour calculer l'énergie consommée.

5.3.2 Analyse de l'impact de la profondeur

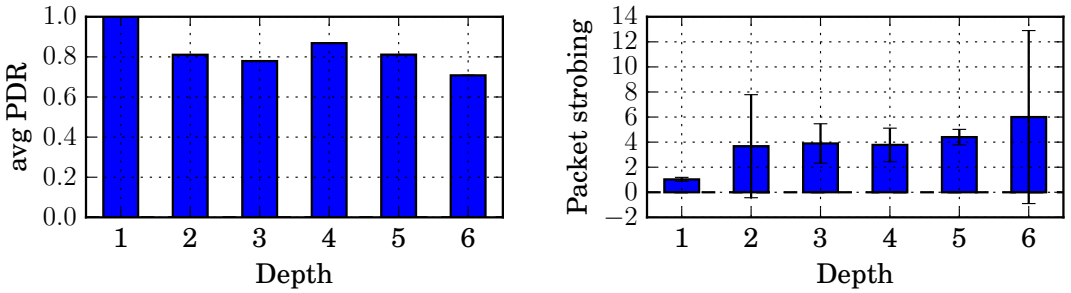
La figure Fig. 5.4a représente le ratio de succès de transmission de paquets pour chaque nœuds en fonction de la distance à la racine. Nous pouvons voir que le ratio de paquets acheminés n'est

Tension d'alimentation	3.6 V
MCU on, Radio RX	21.8 mA
MCU on, Radio TX	19.5 mA
MCU on, Radio off	1800 μA
MCU standby	5.1 μA
MCU idle, Radio off	54.5 μA

TABLE 5.1 – Consommation énergétique du Tmote sky

pas uniforme sur tout le réseau et que les nœuds qui ne sont pas connectés directement à une racine souffrent de congestions et de collisions. La figure 5.4a nous révèle la proportion du trafic que la passerelle peut effectivement voir.

Les nœuds les plus proches d'une racine doivent relayer plus de trafic en provenance des nœuds sous-jacents en plus de leurs propres trafic. De plus les pertes de paquets sont fréquentes notamment loin de la passerelle. Ces pertes de paquets sont causées par des congestions fréquentes dans une topologie comme celle de la chaine.



(a) Packet Delivery Ratio (PDR).

(b) Tentatives d'envois de paquets par profondeur.

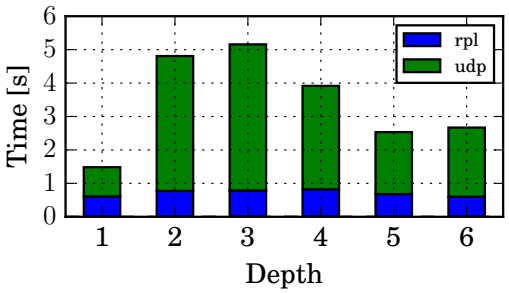
FIGURE 5.4 – Impact de la profondeur

La figure 5.4b représente le nombre moyen d'envois de paquets i.e. le nombre de tentatives que vont faire chaque nœud pour envoyer une trame à son parent. Nous remarquons une corrélation nette entre le nombre de sauts à la distance à la racine et cette courbe. Nous en déduisons que plus un noeud est loin de la racine, plus son nombre de retransmissions va être élevé et ainsi sa consommation énergétique sera plus grande.

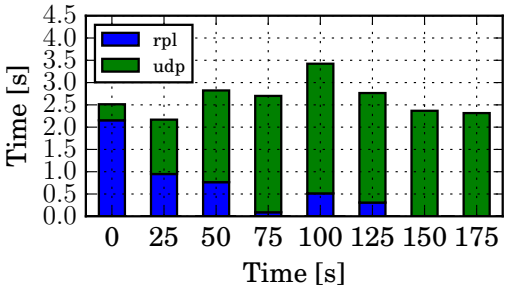
5.3.3 Répartition et évolution des protocoles

La figure. 5.5a représente la distribution des types de trafic se présentant sur chaque saut (Paquets de routage et UDP). Ces mesures montre l'importance du contrôle de trafic que la passerelle ne mesure pas et devraient inférer. Nous remarquons qu'à ce stade, le trafic est presque identique pour tous les nœuds et pourrait être modélisé par un flux constant.

Cependant, la figure 5.5b montre l'évolution de ce phénomène. Nous pouvons observer un gros volume de paquets de routage est requis pour construire le DODAG utilié par RPL. Une fois cette



(a) Répartition des protocoles par profondeur.



(b) Évolution de la répartition des protocoles.

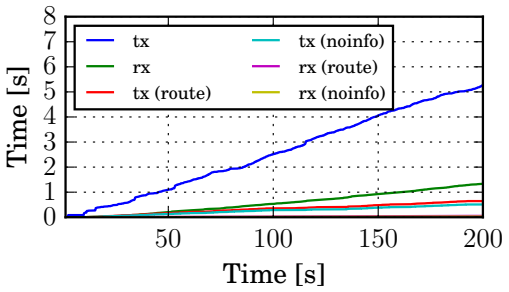
FIGURE 5.5 – Impact des protocoles

phase terminée, le mécanisme de Trickle permet de réduire la quantité de paquet de routage émis car le réseau est stabilisé ; la majorité du trafic devient applicative.

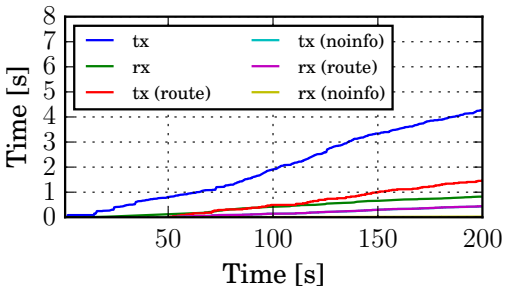
Cela confirme que la construction et la réparation des structures de routage génèrent un large trafic difficile à inférer car local et soumis aux retransmissions. Cependant la pertinence de l'estimation basé sur le trafic applicatif est bien motivé lorsque le réseau est dans un état stable.

5.3.4 Précision de la supervision passive

La figure **??** représente le rapport entre le temps estimé et le temps réellement passé en transmission et réception pour les nœuds 3 et 4.



(a) Supervision passive pour le noeud 3



(b) Supervision passive pour le noeud 5

La consommation énergétique estimé étant une somme linéaire sur la transmission et la réception, nous avons jugé plus pertinent d'exhiber le détail des estimations de temps de transmissions et de réceptions dans nos résultats.

Les nœuds 3 et 5 relaie une part importante de trafic des nœuds sous jacents.

Le nœud 7 ne relaie aucun paquet ainsi le temps passé en transmission et en réception est beaucoup plus faible que pour les autres noeuds.

Nous pouvons voir dans tous les cas que l'estimateur ne disposant d'aucune information de routage (no info) dans les légendes sous-estimes largement l'activité des nœuds. Ce phénomène est attendu car les couts de retransmissions des nœuds intermédiaires sont ignorés en utilisant cette méthode. La topologie n'étant pas en étoile, les différences sont donc importantes.

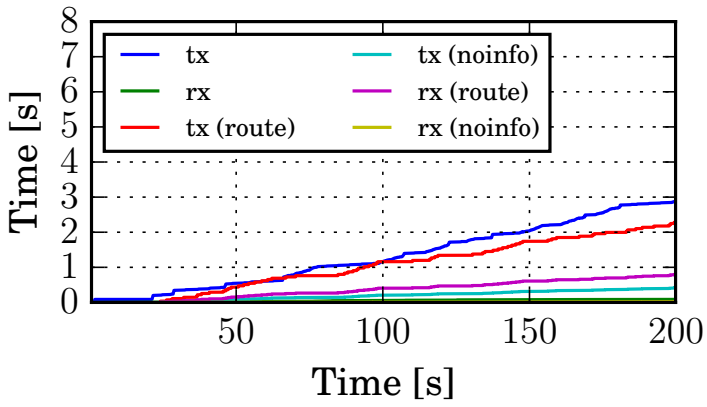


FIGURE 5.7 – Estimation pour un noeud “feuille”

Lorsque la connaissance des routes est utilisé l'estimation est meilleure mais toujours en dessous des valeurs réelles. La sous évaluation initiale est expliquée par la phase de construction du DODAG comme illustré sur la figure 5.5a.

Cependant les différences restent non négligeables même après que la topologie de routage ait été établis ce qui signifie que l'estimation passive n'est pas suffisante, un mécanisme de correction reposant sur des valeurs réelles est nécessaire pour avoir une supervision efficace.

5.3.5 Supervision active et fréquence de correction

Nous évaluons le processus de supervision active en considérant une topologie réseau représentée sur la figure 5.8. Les messages de supervision sont envoyés par les nœuds pour économiser le cout d'une requête par un mécanisme de publish-subscribe dont la fréquence d'envois de notification est réglée par l'utilisateur. Cette topologie est composée de 21 nœuds clients envoyant des paquets à la racine chaque seconde.

Nous avons pris la valeur de $\alpha = 0.25$ afin de ne pas réagir trop brusquement aux pics de trafic brusques causés par une reconstruction du DODAG RPL qui sont courants lorsque ContikiMAC est utilisé.

Il peut être calculé en utilisant une Exponentially Weighted Moving Average (EWMA), comme montré dans l'équation 5.4. Au temps t t, cette erreur est prise proportionnellement au temps depuis la dernière supervision active qui se produit tous les T.

TODO : Ajouter une mention à l'équation explicitement.

Comme nous pouvons le voir sur la figure 5.9, la supervision active périodique fournit la mesure correcte. Malgré la divergence observée au début du à la construction des structures de routage comme expliqué dans la section 5.3.1.

Les messages de supervision active améliorent la précision de l'estimation et permettent d'apprendre le biais de la supervision passive. Ils ont cependant un cout linéaire avec les intervalles de de supervision active et devrait être aussi réduit que possible. Fig. 5.10 représente le ratio moyen obtenu par l'estimateur utilisant la topologie de routage *Route* sur une simulation de 200 seconds en fonction des périodes entre chaque messages de supervision active.

Nous observons que les estimations de transmission divergent beaucoup plus que les estimations de réception. C'est cohérent avec les résultats montrés dans la figure 5.2. La couche MAC est asyn-

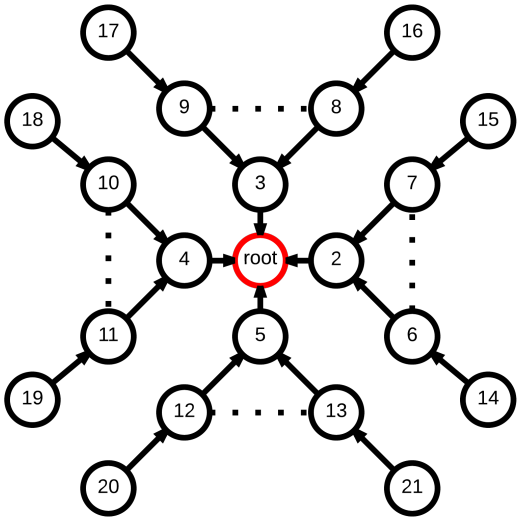


FIGURE 5.8 – Topologie réseau et radio.

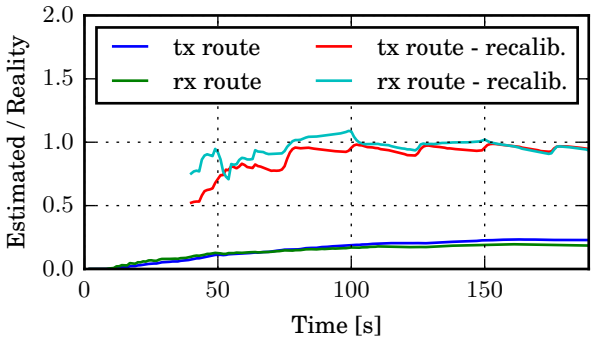


FIGURE 5.9 – Erreur relative pour la topologie réseau avec une supervision active ($T = 25s$).

chrone, ainsi les nœuds passent plus de temps à transmettre qu'à recevoir. Ainsi les biais d'estimation ont plus d'impact pour la transmission que pour la réception.

Afin de réduire le cout des messages de supervision devrait être réduit au cours du temps à mesure que le réseau se stabilise.

5.3.6 Discussion sur la supervision active & passive

La supervision passive construit un modèle de fonctionnement du LLN. Ce modèle peut être mis en défaut lorsque le LLN évolue ou que les noeuds modifient leur fonctionnement. Cependant si la passerelle ne peut savoir si un noeud a changé, comment décider quel noeud doit être intérrogé ? Choisir entre conserver un modèle incertain et ne pas solliciter un noeud ou bien payer le cout d'une supervision active quitte à ce qu'elle ne soit pas pertinente est une problématique de "exploration vs. exploitation" [76].

Dans la mesure où la distribution de l'incertitude n'est pas constante dans le temps, ni connue à l'avance, les approches de restless bandits classiques seront très complexes et d'autres modélisations

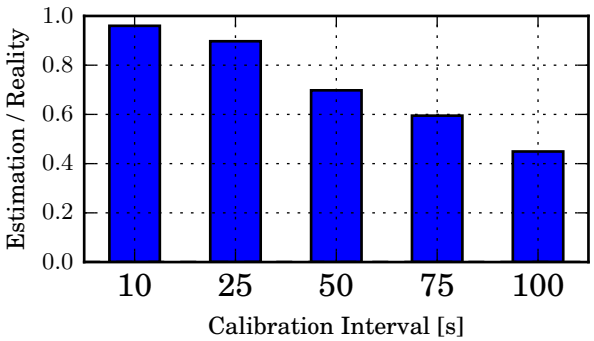


FIGURE 5.10 – Erreur relative pour différents intervalles de supervision active.

peuvent émerger.

Des approches venant du monde de la détection d'anomalies [76] et utilisant des Restless Multi-Armed Bandit (RMAB) sont une piste intéressante pour explorer cette voie.

Cependant une formalisation en utilisant des RMAB nécessite une définition du cout de laisser un noeud sans supervision qui n'est pas trivial à définir dans un cas de supervision.

5.4 Conclusion

Dans ce chapitre, nous avons introduit un mécanisme de supervision passive fonctionnant au niveau de la passerelle, utilisant le trafic qui y passe pour évaluer la consommation énergétique individuelle des nœuds.

Nous montrons, à travers des simulations effectuées avec COOJA, que l'utilisation d'informations déjà disponibles à la passerelle permet d'améliorer la précision des estimateurs de trafic. Cependant, prendre en compte seulement le trafic passant par la passerelle ne suffit pas pour avoir une estimation précise du trafic interne au réseau. Un mécanisme de supervision active est nécessaire pour tenir compte des multiples phénomènes qui ne peuvent être inférés comme les cycles d'endormissement et les transmissions multiples de trames qui leur sont dus.

Ces techniques de supervision passives devraient gagner en précision quand elles sont utilisées avec des couche MAC comme Time-Slotted Channel Hopping (TSCH) dans lequel la passerelle joue un rôle encore plus prépondérrant et où le trafic pour chaque noeud est connu.

La supervision passive ajoute lorsque les déploiements ne le permettent pas des possibilités d'inférence de l'état des noeuds en fonction de phénomènes tiers comme leur trafic réseau qui sont observables au niveau de la passerelle. Cependant aussi sophistiquée que soit l'inférence, une vérification des hypothèses et des modèles demeurrera toujours nécessaire rendant la supervision active indispensable. La supervision passive permet de réduire le cout de la supervision active en faisant un compromis entre la granularité de la supervision et son cout énergétique.

Publications

Rémy Léone, Jérémie Leguay, Paolo Medagliani, Claude Chaudet. Tee : Traffic-based energy estimators for duty-cycled Wireless Sensor Networks. *IEEE International Conference on Communication (ICC)*, page 6749-6754, Londres, 2015.

Chapitre 6

Conclusion & Perspectives

6.1 LLNs pour l'Internet des Objets

Les problématiques apportées par les LLN nous permettent d'étendre l'Internet a un plus large ensemble pour construire l'Internet du futur.

6.2 Connexion des LLNs à l'Internet : Nécessité d'une passerelle avancée

6.2.1 Contributions

Comme nous l'avons vu tout au long de cette thèse, les passerelles peuvent et sont utilisés pour de multiples usages. En plus des fonctionnalités basiques de connexion et de sécurité. D'autres fonctions réseaux peuvent y être implémentée.

Les fonctionnalités de cache

Les fonctionnalités de supervision Dans le futur, il est possible que l'ensemble des fonctionnalités présentées soient présentes à un niveau complètement virtualisé sous la forme d'une Network Function Virtualization (NFV).

6.2.2 Ouvertures

- Méthodes de changements de temps de validité de requêtes utilisant la popularité d'un contenu en plus des paramètres d'énergies - Interopérabilité entre plus de protocoles - Modèle de supervision passif sur des protocoles plus déterministe (6TiSCH)

6.3 Reproductibilité en simulations et expérimentations

Cependant la variété et la taille de ces réseaux imposent des méthodes systématiques de tests de fonctionnement et d'interopérabilité.

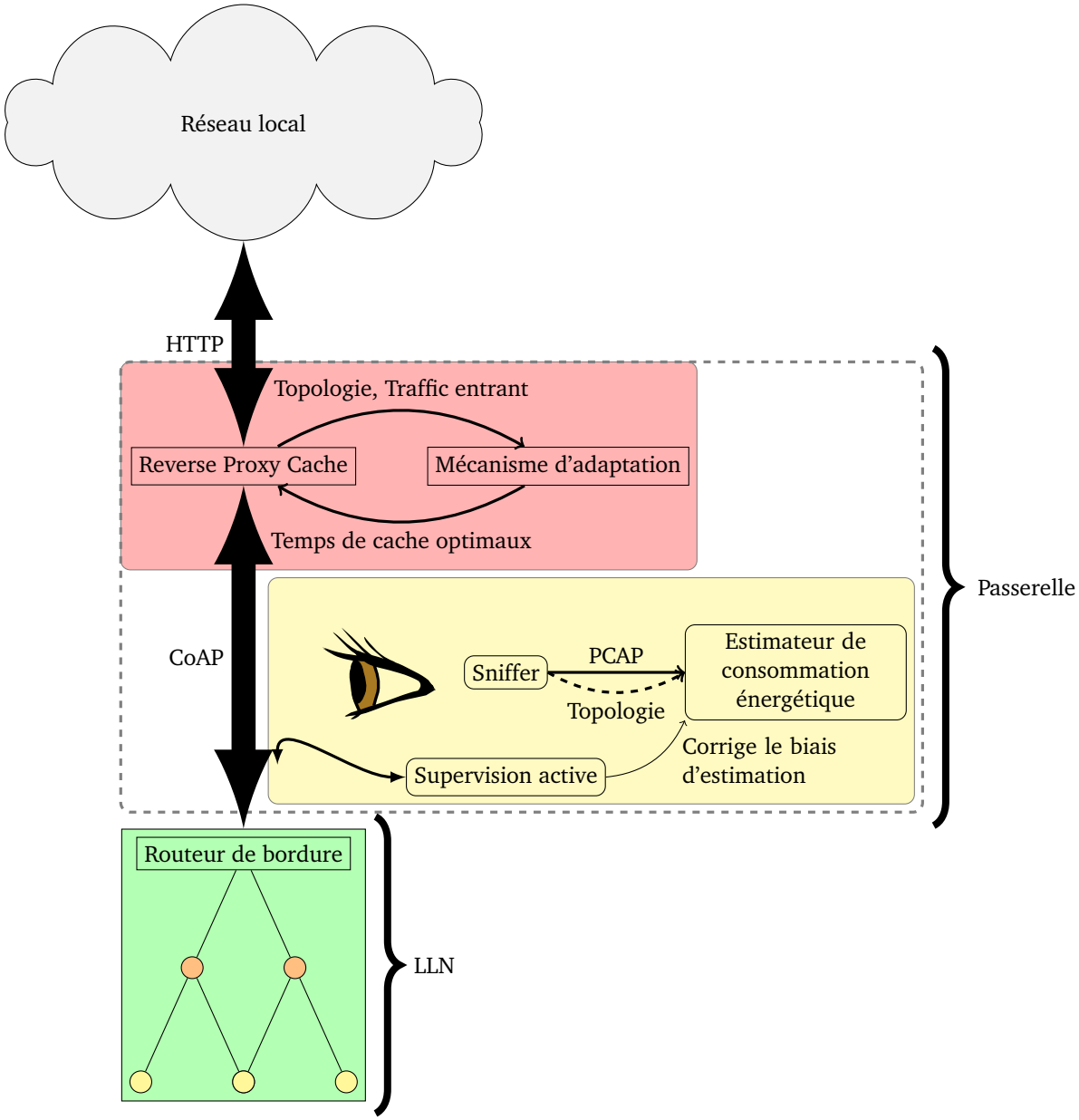


FIGURE 6.1 – Schéma de la passerelle proposée

6.3.1 Contributions

- Documentation d'une expérience - Automatisation d'une partie des tâches

6.3.2 Ouvertures

- Gestion des événements au runtime - Utilisation de plateformes fédérés pour le déploiements de tests et d'expériences

Annexe A

Analyse énergétique et modélisation du réseau avec ContikiMAC

TODO : Il faut également vérifier que les unités sont cohérentes. TODO : Offrir une formule complète à la fin.

Nous proposons une analyse théorique de la consommation d'un noeud utilisant ContikiMAC. Le modèle étant complexe dans le cas général, nous avons donc développé un calcul approché de la consommation énergétique qui intègre la topologie du réseau.

L'utilisation de l'interface radio étant le principal consommateur de la batterie dans nos simulations, nous avons analysé ce protocole afin d'estimer la durée de vie du réseau.

A.1 Introduction

ContikiMAC est un mécanisme de cycle de veille disponible dans Contiki-OS [34]. Similaire à BoX-MAC [87], il est en charge d'éteindre et d'allumer les interfaces radio d'un nœud dans le but d'économiser de l'énergie. Ce mécanisme de cycle de veille construit au dessus de IEEE 802.15.4 permet au nœud d'éteindre et d'allumer sa radio périodiquement [1] pour consulter l'activité du canal.

Les réveils de nœuds voisins peuvent être désynchronisés, ainsi la source d'une trame devra l'envoyer à plusieurs reprises (*strobing*) jusqu'à la réception d'un acquittement ou bien le nombre maximal de retransmissions possibles. Lors d'un réveil, si un nœud détecte que le canal est occupé, il reste éveillé jusqu'à la réception de la trame entière. Si ce nœud est le destinataire de cette trame il reste éveillé pour le recevoir entièrement sinon il éteint sa radio. Des mécanismes complémentaires de phase-locking sont mis en place afin de réduire les envois répétés de paquets par l'expéditeur.

Ce mécanisme d'endormissement n'est généralement pas mis en place au niveau de la passerelle car il n'est pas souhaitable d'avoir des délais importants pour contacter la sortie du réseau, notamment dans des schémas de communications multipoint to point. De plus, la passerelle assure une connexion permanente à un réseau local conventionnel et est un nœud non-contraint énergétiquement.

1. par défaut toutes les 125 ms

A.2 États de transmission d'un noeud

Il y a 4 états possibles pour un nœud utilisant ContikiMAC : (i) transmission, (ii) réception, (iii) mode sommeil et (iv) écoute du canal avec les consommations énergétiques suivantes : $\Omega_{\mathscr{T}}$, $\Omega_{\mathscr{R}}$, $\Omega_{\mathscr{S}}$, et $\Omega_{\mathscr{L}}$ (dimension :[W]), respectivement.

$$\Omega = \Omega_{\mathscr{S}} + \Omega_{\mathscr{L}} + \Omega_{\mathscr{T}} + \Omega_{\mathscr{R}} \tag{A.1}$$

où : $\Omega_{\mathscr{S}}$ est l'énergie consommée pendant la période de sommeil du i-ème nœud ; $\Omega_{\mathscr{L}}$ est l'énergie demandée pour écouter le canal sur une période fixée ; $\Omega_{\mathscr{R}}$ est l'énergie utilisée pour la réception d'un paquet ; $\Omega_{\mathscr{T}}$ est l'énergie utilisée pour transmettre un paquet.

A.3 Énergie résiduelle

Nous pouvons dire que la consommation énergétique de chaque nœud est donnée par la somme des consommations de ses composants. L'énergie résiduelle d'un nœud i à un instant t peut être exprimé par :

$$E_r(t) = E_0 - \Omega t \tag{A.2}$$

E_0 est l'énergie initiale du nœud considéré et Ω (dimension : [W]) est la puissance consommée par le système.

A.4 Temps de transmission & réception

Nous définissons le temps T_C comme la durée entre deux phases actives consécutives et $T_{\mathscr{S}}$ comme la durée de la phase de sommeil du noeud considéré. La durée de la phase active peut être évaluée comme : $T_{A_i} = T_{C_i} - T_{\mathscr{S}_i}$.

Nous concentrerons notre étude sur un réseau offrant du trafic applicatif CoAP avec des paquets fixes.

Il existe 3 types de nœuds dans notre modèle : (i) Les serveurs CoAP, (ii) les routeurs qui agissent aussi comme serveurs CoAP et (iii) une racine du DODAG. Puisque la taille d'une requête et d'une réponse ont des tailles différentes, nous distinguerons le temps pour transmettre une requête : $T_r = L_r/R$ de celui de la réponse défini comme $T_a = \frac{L_a}{R}$, où L_r et L_a sont les tailles de paquet de la requête et de la réponse respectivement et R est le débit de transmission des nœuds. D'après le protocole Contiki MAC [34] quand un nœud transmet un paquet, il reste durant $T_{p,\mathscr{T}\to\mathscr{T}}$ en transmission et pour une période $T_{p,\mathscr{T}\to\mathscr{R}}$ en réception afin de recevoir le paquet ACK du destinataire. Ces périodes de temps peuvent être exprimée par :

$$T_{p,\mathscr{T}\to\mathscr{T}} = \frac{3 + \lfloor \frac{T_{\mathscr{S}} - T_p}{T_p} \rfloor}{2} T_p \tag{A.3}$$

$$T_{p,\mathscr{T}\to\mathscr{R}} = \frac{3 + \lfloor \frac{T_{\mathscr{S}} - T_p}{S_p} \rfloor}{2} T_d + T_{\mathscr{A}} \tag{A.4}$$

où T_p indique un paquet générique qu'il soit une requête T_r ou une réponse T_a. T_d est le temps requis pour détecter avec succès un acquittement d'un récepteur, et $T_{\mathscr{A}}$ est le temps nécessaire pour

transmettre un ACK. L'équation A.3 est obtenue en faisant la moyenne entre le meilleur cas de transmission (c'est-à-dire quand le nœud commence à transmettre alors que le destinataire vient de se réveiller), et le pire cas (Le destinataire vient de rentrer en sommeil alors que le nœud commence à transmettre). De la même façon, quand un nœud reçoit un paquet, il passe une partie de son temps en réception et une partie de son temps en transmission puisqu'il a besoin de transmettre un ACK à l'expéditeur du paquet. Ces périodes de temps peuvent être exprimées par :

$$T_{p,\mathscr{R}\to\mathscr{R}} = \frac{3T_p}{2} + T_p \tag{A.5}$$

$$T_{p,\mathscr{R}\to\mathscr{T}} = T_{\mathscr{A}} \tag{A.6}$$

où T_p est l'intervalle entre chaque transmission de paquet $T_{\mathscr{A}}$ est la durée de transmission d'un paquet ACK. L'équation (A.5) est obtenue en faisant la moyenne entre le meilleur et le pire des cas. Le meilleur étant quand le paquet n'a besoin d'être transmis qu'une fois et le pire qui est celui où le nœud se réveille juste après le début d'une transmission par l'expéditeur. Dans ce cas, le nœud doit attendre la prochaine transmission pour tenter de le recevoir correctement.

A.5 Consommation dues aux transmissions applicatives

Les termes $\Omega_{\mathscr{T}}$ et $\Omega_{\mathscr{R}}$ vont être différents selon le nœud visé. Nous distinguons trois cas, celui d'un serveur CoAP simple ($\Omega_{server,\mathscr{T}}$ et $\Omega_{server,\mathscr{R}}$, d'un routeur ($\Omega_{router,\mathscr{T}}$ et $\Omega_{router,\mathscr{R}}$) ou bien de la racine du DODAG ($\Omega_{root,\mathscr{T}}$ et $\Omega_{root,\mathscr{R}}$). Nous supposerons que le temps moyen entre deux requêtes consécutives pour un noeud est de r secondes.

A.5.1 Consommation des serveurs applicatifs simples

Dans le cas d'un serveur applicatif CoAP qui ne fait que recevoir une requete et transmet une valeur observée, la puissance consommée durant la transmission et la réception peut alors être exprimée par :

$$\Omega_{server,\mathscr{T}} = \frac{P_{\mathscr{T}} T_{a,\mathscr{T}\to\mathscr{T}} + P_{\mathscr{R}} T_{a,\mathscr{T}\to\mathscr{R}}}{r} \tag{A.7}$$

$$\Omega_{server,\mathscr{R}} = \frac{P_{\mathscr{R}} T_{r,\mathscr{R}\to\mathscr{R}} + P_{\mathscr{T}} T_{r,\mathscr{R}\to\mathscr{T}}}{r} \tag{A.8}$$

en replacant les tailles de paquets dans les requêtes et leurs réponses respectives dans les équations (A.3), (A.4), (A.5), et (A.6). Les termes $P_{\mathscr{T}}$ et $P_{\mathscr{R}}$ désignent la puissance consommée par un nœud en phase de réception et de transmission.

A.5.2 Consommation de la racine du DODAG

Dans le cas d'une racine, la puissance consommée pour transmettre à un noeud générique i et recevoir sa réponse peut être exprimée par :

$$\Omega_{root,\mathscr{T}} = \frac{P_{\mathscr{T}} T_{r,\mathscr{T}\to\mathscr{T}} + P_{\mathscr{R}} T_{r,\mathscr{T}\to\mathscr{R}}}{r} \tag{A.9}$$

$$\Omega_{root,\mathscr{R}} = \frac{P_{\mathscr{R}} T_{a,\mathscr{R}\to\mathscr{R}} + P_{\mathscr{T}} T_{a,\mathscr{R}\to\mathscr{T}}}{r} \tag{A.10}$$

Puisque la racine transmet à chacun de ces enfants, la puissance consommée pour transmettre à tous les nœuds et recevoir peut être exprimée par :

$$\Omega_{root,\mathscr{T}} = \sum_{i=1}^{N} \Omega_{root,\mathscr{T}} \tag{A.11}$$

$$\Omega_{root,\mathscr{R}} = \sum_{i=1}^{N} \Omega_{root,\mathscr{R}} \tag{A.12}$$

TODO : Attention aux indices

A.5.3 Consommation des nœuds relais/serveurs

Les routeurs répondent aux requêtes qui les concernent mais servent aussi d'intermédiaires celles à destination de leurs enfants. Ainsi la puissance consommée pour transmettre les paquets peut être exprimée par :

$$\Omega_{router,\mathscr{T}} = \sum_{j \in m_i} \left(\Omega_{server,\mathscr{T}_j} + \Omega_{root-T_j} \right) + \Omega_{server,\mathscr{T}} \tag{A.13}$$

$$\Omega_{router,\mathscr{R}} = \sum_{j \in m_i} \left(\Omega_{server,\mathscr{R}_j} + \Omega_{root-R_j} \right) + \Omega_{server,\mathscr{R}} \tag{A.14}$$

où m_i désigne l'ensemble des enfants du nœud i. Les termes à la droite de (A.13) et (A.14) sont introduits car ils peuvent aussi répondre eux-mêmes à des requêtes applicatives.

A.6 Consommation en phase d'écoute de canal & sommeil

Enfin, la puissance consommée en écoute de canal et dans l'état de sommeil peut être exprimée par :

$$\Omega_{\mathscr{L}} = \frac{T_{\text{fl}} P_{\mathscr{R}}}{T_C} \tag{A.15}$$

$$\Omega_{\mathscr{S}} = \frac{T_{\mathscr{S}} P_{\mathscr{S}}}{T_C} - \Gamma_{\mathscr{T}} - \Gamma_{\mathscr{R}} \tag{A.16}$$

où $P_{\mathscr{S}}$ est la puissance durant l'état de sommeil, $\Gamma_{\mathscr{T}}$ et $\Gamma_{\mathscr{R}}$ sont deux termes correctifs. En temps normal, un nœud effectue soit une écoute du canal, soit transmet ou reçoit un paquet soit dort. $\Gamma_{\mathscr{T}}$ et $\Gamma_{\mathscr{R}}$ sont utilisés pour raffiner la puissance consommée durant la phase de sommeil. Les périodes de sommeils chevauchent celles de transmissions ou réceptions sur des temps courts ainsi sans ces termes la puissance consommée sera surestimée. $\Gamma_{\mathscr{T}}$ et $\Gamma_{\mathscr{R}}$ peuvent être exprimée par :

$$\Gamma_{\mathscr{T}} = \Omega_{\mathscr{S}} \frac{1}{T_C} \left(\frac{3 + \lfloor \frac{T_{\mathscr{S}_i} - T_p}{T_p} \rfloor}{2} (T_p + T_d) + T_{\mathscr{A}} \right) \tag{A.17}$$

$$\Gamma_{\mathscr{R}} = \Omega_{\mathscr{S}} \frac{1}{T_C} \left(\frac{3T_p}{2} + T_p + T_{\mathscr{A}} \right) \tag{A.18}$$

Le terme $\Gamma_{\mathscr{T}}$ tient du fait que durant les opérations de transmissions comme les phases de (i) transmissions stroboscopiques d'un paquet sur une phase $T_{\mathscr{S}}$, (ii) la transmission standard d'un paquet et (iii) la réception d'un acquittement, un nœud serait normalement en sommeil. Ainsi le facteur correctif $\Gamma_{\mathscr{T}}$ est nécessaire puisque autrement l'énergie consommée par un nœud avec ce modèle serait surestimée car réception et transmission se chevaucherait avec les opérations de sommeil normales sur une période. Des considérations similaires peuvent être dressées pour le terme $\Gamma_{\mathscr{R}}$. Quand un nœud attends un acquittement pour transmettre un paquet ACK, pour recevoir le préambule et le paquet, le nœud serait normalement en état de sommeil.

En injectant les expressions (A.17) et (A.18) dans (A.16) et les expressions (A.7), (A.8) (si il s'agit d'un noeud CoAP autrement (A.13) et (A.14) pour un routeur ou (A.11) et (A.12) pour la racine), (A.15), et (A.16) dans (A.1), il est possible de dériver une expression pour la consommation qui ne dépends que de la topologie et des paramètres de communications.

Annexe B

Extraits de code source utilisés par Makesense

B.1 Fabrication

```
with open(pj(path, "main.csc"), "w") as f:
    f.write(main_csc_template.render(
        title="Dummy Simulation",
        random_seed=12345,
        transmitting_range=42,
        interference_range=42,
        success_ratio_tx=1.0,
        success_ratio_rx=1.0,
        mote_types=[
            {"name": "server", "description": "server",
                "firmware": "dummy-server.wismote"},
            {"name": "client", "description": "client",
                "firmware": "dummy-client.wismote"}
        ],
        motes=[
            {"mote_id": 1, "x": 0, "y": 0, "z": 0, "mote_type": "server"},
            {"mote_id": 2, "x": 1, "y": 1, "z": 0, "mote_type": "client"},
        ],
        script=script))
```

```
<?xml version="1.0" encoding="UTF-8"?>
<simconf>
  <simulation>
    <title>{{ title }}</title>
    <randomseed>{{ random_seed }}</randomseed>
    <radiomedium>
      org.contikios.cooja.radiomediums.UDGM
      <transmitting_range>{{ transmitting_range }}</transmitting_range>
```

```
      <interference_range>{{ interference_range }}</interference_range>
      <success_ratio_tx>{{ success_ratio_tx }}</success_ratio_tx>
      <success_ratio_rx>{{ success_ratio_rx }}</success_ratio_rx>
    </radiomedium>
    {% for mote_type in mote_types %}
    <motetype>
      org.contikios.cooja.mspmote.WismoteMoteType
      <identifier>{{ mote_type.name }}</identifier>
      <firmware EXPORT="copy">{{ mote_type.firmware }}</firmware>
    </motetype>
    {% endfor %}
    {% for mote in motes %}
    <mote>
      <interface_config>
        org.contikios.cooja.interfaces.Position
        <x>{{ mote.x }}</x>
        <y>{{ mote.y }}</y>
        <z>{{ mote.z }}</z>
      </interface_config>
      <interface_config>
        org.contikios.cooja.mspmote.interfaces.MspMoteID
        <id>{{ mote.mote_id }}</id>
      </interface_config>
      <motetype_identifier>{{ mote.mote_type }}</motetype_identifier>
    </mote>
    {% endfor %}
  </simulation>
  <plugin>
    org.contikios.cooja.plugins.ScriptRunner
    <plugin_config>
      <script>
      {{ script }}
      </script>
    </plugin_config>
  </plugin>
</simconf>
```

Nous pouvons voir des cas de paramètrages simples avec un remplacement clé → valeur.

Il est également possible d'avoir des boucles, des itérations et des conditionnelles. C'est le cas par exemple dans la création des emplacements des noeuds introduits grace à la boucle `for`.

Notons ici que la balise `script` utilise également un moteur de templating. Ainsi, la configuration du programme qui peut lui aussi contenir des conditionnels et des boucles d'execution peut être aussi géré avec ce mécanisme.

B.2 Déploiement

```
import fabric

@host("grenoble")
def run():
        run_experiment
```

B.3 Parsing

```
import subprocess
from os.path import join as pj

def pcap2csv(folder, filename="output.csv"):
    """
    Execute a simple filter on PCAP and count
    """
    # Getting raw data
    with open(pj(folder, filename), "w") as f:

        command = ["tshark",
                   "-T", "fields",
                   "-E", "header=y",
                   "-E", "separator=,",
                   "-Y", "udp || icmpv6",
                   "-e", "frame.time_epoch",
                   "-e", "frame.protocols",
                   "-e", "frame.len",
                   "-e", "wpan.fcs",
                   "-e", "wpan.seq_no",
                   "-e", "wpan.src16",
                   "-e", "wpan.dst16",
                   "-e", "wpan.src64",
                   "-e", "wpan.dst64",
                   "-e", "icmpv6.type",
                   "-e", "ipv6.src",
                   "-e", "ipv6.dst",
                   "-e", "icmpv6.code",
                   "-e", "udp.dstport",
                   "-e", "udp.srcport",
                   "-e", "data.data",
                   "-r", pj(folder, "output.pcap")]

        process = subprocess.Popen(command, stdout=subprocess.PIPE)
        stdout, stderr = process.communicate()
        f.write(stdout)
```

Afin d'avoir un traitement des données aisé, il est préférable de se ramener à des formats de fichiers textuels tel que le CSV. Ici, tshark est utilisé comme intermédiaire pour traduire un format binaire et extraire les informations les plus essentielles vers du texte.

Il est à noté que dès que cette transformation a été faite une fois, elle n'est jamais effectuée. En effet, le fait d'avoir sauvegardé les données dans un format intermédiaire permet de ne travailler qu'avec le CSV et de ne plus jamais avoir à faire une extraction d'information couteuse et redondante. De plus dans le cas d'un fichier binaire au format changeant, avoir une version texte garantis que ces données pourront être lus postérieurement si le traducteur (en l'occurence tshark) n'est plus disponible.

B.4 Analyse

```
import pandas as pd

df = pd.read_csv("my_results.csv")
df[df.pkt_type == "udp"].count()
```

B.4.1 Filtrage

Nous obtenons ainsi en une ligne de code, un graphe représentant l'histogramme du nombre de paquets.

B.4.2 Fonctions aggrégées

Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.

B.4.3 Traitements en masse

Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.

B.5 Présentation

```
import matplotlib.pyplot as plt
import pandas as pd

df = pd.read_csv("my_results.csv")
df[df.pkt_type == "udp"].count().plot(kind="bar")
```

B.6 Intégration continue (Travis-ci)

```
language: python

# Mise en place de l'environnement
install:
    # La compilation des sources de la suite python peut être un
    # peu longue. Il est commode d'utiliser des paquets binaires afin
    # d'avoir un build rapide
  - "sudo apt-get -qq install ipython"
  - "sudo apt-get -qq install python-matplotlib"
  - "sudo apt-get -qq install fabric"
  - "sudo apt-get -qq install python-pandas"
  - "sudo apt-get -qq install python-networkx"
  - "sudo apt-get -qq install python-numpy"
  - "sudo apt-get -qq install python-jinja2"
  - "sudo apt-get -qq install python-scipy"

  # Récupération des sources de contiki et des compilateurs nécessaires
  - "git clone --recursive git://github.com/contiki-os/contiki"
  - sudo apt-get -qq update
  - sudo apt-get -qq install lib32z1
  - wget http://simonduq.github.io/resources/mspgcc-4.7.2-compiled.tar.bz2 &&
    tar xjf mspgcc*.tar.bz2 -C /tmp/ &&
    sudo cp -f -r /tmp/msp430/* /usr/local/ &&
    rm -rf /tmp/msp430 mspgcc*.tar.bz2 &&
    msp430-gcc --version

    # Utile pour analyser les traces PCAP
  - "sudo apt-get install tshark"

# Execution. Si ces commandes renvoient 0
# le test est considéré comme un succès
script:
  - pip install -r requirements.txt
  # Conversion du notebook vers un script executable
  - "ipython nbconvert --to=python demo.ipynb"
  # Execution du script
  - python demo.py
```

L'utilisation de Travis-ci peut être justifié par le fait qu'un tiers à priori non lié à une organisation ou un organisme de recherche peut être de bonne foi quant au caractère reproductible. L'intégration continue garantis que l'expérience pourra être reproduite. Un point important de cette configuration est que les logiciels sont spécifiés avec une version (requierements.txt). Ainsi, au cas où la bibliothèque viendrait à ne plus être maintenue ou comporterait des changements d'interfaces, il serait toujours possible de retrouver d'anciennes versions et reproduire l'expérience avant de la faire migrer vers de nouvelles versions.

Annexe C

Collaborations extérieures

C.1 A scalable and self-configuring architecture for service discovery in the internet of things

Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.

C.2 Bounding Degrees on RPL

Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.

C.3 Tactique de supervision active économe en énergie

Collaboration avec Simon en suède sur le bandit manchot
Mettre quelques graphes.

Acronymes

h Cache Hit ratio
m Cache Miss ratio
6LBR 6LoWPAN Border Router
6LoWPAN IPv6 over Low power Wireless Personal Area Networks
ACK Acknowledgement message
API Application Programming Interface
ARP Address Resolution Protocol
BAN Body-Area Network
BLE Bluetooth Low-Energy
CBOR Concise Binary Object Representation
CoAP Constrained Application Protocol
CON Confirmable message
CoRE Constrained RESTful environment
CPL Courants Porteurs en Ligne
CSMA/CA Carrier Sense Multiple Access with Collision Avoidance
CSV Comma Separated Values
DAD Duplicate Address Detection
DAG Directed Acyclic Graph
DAO Destination Advertisement Object
DIO DODAG Information Object
DIS DODAG Informational Solicitation
DHCP Dynamic Host Configuration Protocol
DODAG Destination-Oriented DAG
DNS Domain Name System
DSL Domain Specific Language
DTLS Datagram Transport Layer Security
EWMA Exponentially Weighted Moving Average

ETag Entity Tag
FFD Full Function Device
HCP HTTP-CoAP proxy
HTTP HyperText Transfer Protocol
ICMPv6 Internet Control Message Protocol v6
IEEE Institute of Electrical and Electronics Engineers
IETF Internet Engineering Task Force
IGP Interior Gateway Protocol
IHM Interaction Homme Machine
IID Interface ID
IoT Internet of Things
IP Internet Protocol
ISM (Industrial, Scientific and Médical
KP Knapsack Problem
JSON Javascript Serial Object Notation
LAN Local Area Network
LBR LoWPAN Border Router
LLN Low-Power and Lossy Networks
LoWPAN Low-Power Wireless Personal Area Network
LPWAN Low-Power Wide Area Network
LRWPAN Low Rate Wireless Personal Area Network
M2M Machine to Machine
MAC Media access control
MP2P Multi-point to point
MQTT Message Queuing Telemetry Transport
MTU Maximum transmission unit
NAT Network Address Translation
NDP Neighbor Discovery Protocol
NFV Network Function Virtualization
NON Non-confirmable message
NSGA Non-dominated Sorting Genetic Algorithm
NTP Network Time Protocol
OTA Over The Air
P2MP Point to Multi-point
P2P Point to Point
PCAP Packet CAPture
PoE Power over Ethernet
QoS Quality of Service

RMAB Restless Multi-Armed Bandit
REST REpresentational State Transfer
RFD Reduced Function Device
ROLL Routing Over Low power and Lossy Networks
RPCA Reverse Proxy Cache Adaptatif
RPC Reverse Proxy Cache
RPL Routing Protocol Layer
RST Reset message
SCADA Supervisory Control and Data Acquisition
SOA Service Oriented Architecturei
SQL Structured Query Language
TCP Transport Control Protocol
TDMA Time Division Multiple Access
Tee Traffic Energy Estimator
TSCH Time-Slotted Channel Hopping
TTL Time To Live
UDP User Datagram Protocol
URI Uniform Resource Identifier
URL Uniform Resource Locator
WSN Wireless Sensor Networks
XML Extensible Markup Language

Bibliographie

[1] 6tisch. IPv6 over the TSCH mode of IEEE 802.15.4e. `http://datatracker.ietf.org/wg/6tisch/`.

[2] Cesare Alippi, Giuseppe Anastasi, Cristian Galperti, Francesca Mancini, and Manuel Rove. Adaptive sampling for energy conservation in wireless sensor networks for snow monitoring applications. In *Mobile Adhoc and Sensor Systems, 2007. MASS 2007. IEEE International Conference on*, pages 1–6. IEEE, 2007.

[3] ZigBee Alliance. Zigbee specification, 2006.

[4] D Antolin, N Medrano, and B Calvo. Analysis of the operating life for battery-operated wireless sensor nodes. In *Industrial Electronics Society, IECON 2013-39th Annual Conference of the IEEE*, pages 3883–3886. IEEE, 2013.

[5] Carles Anton-Haro and Mischa Dohler. *Machine-to-machine (M2M) Communications : Architecture, Performance and Applications*. Elsevier, 2014.

[6] Luigi Atzori, Antonio Iera, and Giacomo Morabito. The internet of things : A survey. *Computer networks*, 54(15) :2787–2805, 2010.

[7] Nouha Baccour, Anis Koubaa, Luca Mottola, Marco Antonio Zuniga, Habib Youssef, Carlo Alberto Boano, and Mário Alves. Radio link quality estimation in wireless sensor networks : a survey. *ACM Transactions on Sensor Networks (TOSN)*, 8(4) :34, 2012.

[8] Debasis Bandyopadhyay and Jaydip Sen. Internet of things : Applications and challenges in technology and standardization. *Wireless Personal Communications*, 58(1) :49–69, 2011.

[9] Paolo Baronti, Prashant Pillai, Vince WC Chook, Stefano Chessa, Alberto Gotta, and Y Fun Hu. Wireless sensor networks : A survey on the state of the art and the 802.15. 4 and zigbee standards. *Computer communications*, 30(7) :1655–1695, 2007.

[10] Olaf Bergmann. libcoap : C-implementation of coap. *http ://libcoap.net*, 2012.

[11] Jeff Bezanzon, Stefan Karpinski, Viral Shah, and Alan Edelman. Julia : A fast dynamic language for technical computing. In *Lang.NEXT*, April 2012.

[12] C. Bormann and P. Hoffman. Concise Binary Object Representation (CBOR). RFC 7049 (Proposed Standard), October 2013.

[13] S Brown and CJ Sreenan. Updating software in wireless sensor networks : A survey. *Dept. of Computer Science, National Univ. of Ireland, Maynooth, Tech. Rep*, 2006.

[14] Nevil Brownlee and KC Claffy. Understanding internet traffic streams : dragonflies and tortoises. *Communications Magazine, IEEE*, 40(10) :110–117, 2002.

[15] Tomasz Buchert, Cristian Ruiz, Lucas Nussbaum, and Olivier Richard. A survey of general-purpose experiment management tools for distributed systems. *Future Generation Computer Systems*, 45 :1–12, 2015.

[16] Qing Cao, Ting Yan, John Stankovic, and Tarek Abdelzaher. Analysis of target detection performance for wireless sensor networks. In *Distributed Computing in Sensor Systems*, pages 276–292. Springer, 2005.

[17] Andrea Caragliu, Chiara Del Bo, and Peter Nijkamp. Smart cities in europe. *Journal of urban technology*, 18(2) :65–82, 2011.

[18] Edward Chan and Song Han. Energy efficient residual energy monitoring in wireless sensor networks. *International Journal of Distributed Sensor Networks*, 5(6), 2009.

[19] Bor-rong Chen, Geoffrey Peterson, Geoff Mainland, and Matt Welsh. Livenet : Using passive monitoring to reconstruct sensor network dynamics. In *IEEE/ACM DCOSS*, 2008.

[20] Chipcon. *2.4 GHz IEEE 802.15.4 / ZigBee-ready RF Transceiver*.

[21] Simone Cirani, Luca Davoli, Gianluigi Ferrari, Rémy Léone, Paolo Medagliani, Marco Picone, and Luca Veltri. A scalable and self-configuring architecture for service discovery in the internet of things. 2014.

[22] Thomas Clausen, Ulrich Herberg, and Matthias Philipp. A critical evaluation of the ipv6 routing protocol for low power and lossy networks (rpl). In *Wireless and Mobile Computing, Networking and Communications (WiMob), 2011 IEEE 7th International Conference on*, pages 365–372. IEEE, 2011.

[23] Lorenzo Colitti, Steinar H Gunderson, Erik Kline, and Tiziana Refice. Evaluating ipv6 adoption in the internet. In *Passive and Active Measurement*, pages 141–150. Springer, 2010.

[24] W. Colitti, K. Steenhaut, and N. De Caro. Integrating wireless sensor networks with the web. *Extending the Internet to Low powerand Lossy Networks (IP+ SN 2011)*, 2011.

[25] Walter Colitti, Kris Steenhaut, Niccolo De Caro, Bogdan Buta, and Virgil Dobrota. Rest enabled wireless sensor networks for seamless integration with web applications. In *Mobile Adhoc and Sensor Systems (MASS), 2011 IEEE 8th International Conference on*, pages 867–872. IEEE, 2011.

[26] Matteo Collina, Giovanni Emanuele Corazza, and Alessandro Vanelli-Coralli. Introducing the qest broker : Scaling the iot by bridging mqtt and rest. In *Personal Indoor and Mobile Radio Communications (PIMRC), 2012 IEEE 23rd International Symposium on*, pages 36–41. IEEE, 2012.

[27] Moteiv Corp. Ultra low power IEEE 802.15.4 compliant wireless sensor module.

[28] Bart De Schutter and Bart De Moor. Optimal traffic light control for a single intersection. *European Journal of Control*, 4(3) :260–276, 1998.

[29] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan. A fast and elitist multiobjective genetic algorithm : NSGA-II. *IEEE Trans. on Evolutionary Computation*, 6(2) :182–197, April 2002.

[30] Kalyanmoy Deb and Samir Agrawal. A niched-penalty approach for constraint handling in genetic algorithms. In *Artificial Neural Nets and Genetic Algorithms*, pages 235–243. Springer, 1999.

[31] Enrique J Duarte-Melo and Mingyan Liu. Analysis of energy consumption and lifetime of heterogeneous wireless sensor networks. In *Global Telecommunications Conference, 2002. GLOBECOM'02. IEEE*, volume 1, pages 21–25. IEEE, 2002.

[32] A. Dunkels, B. Gronvall, and T. Voigt. Contiki-a lightweight and flexible operating system for tiny networked sensors. pages 455–462, 2004.

[33] Adam Dunkels. Contiki regression tests : 9 hardware platforms, 4 processor achitectures, 1021 network nodes. http://contiki-os.blogspot.fr/2012/12/contiki-regression-tests-9-hardware.html.

[34] Adam Dunkels. The ContikiMAC Radio Duty Cycling Protocol. Technical Report T2011 :13, Swedish Institute of Computer Science, December 2011.

[35] Simon Duquennoy, Niklas Wirström, Nicolas Tsiftes, and Adam Dunkels. Leveraging ip for sensor network deployment. In *Proceedings of the workshop on Extending the Internet to Low power and Lossy Networks (IP+ SN 2011), Chicago, IL, USA*, volume 11. Citeseer, 2011.

[36] Paul M Duvall, Steve Matyas, and Andrew Glover. *Continuous integration : improving software quality and reducing risk*. Pearson Education, 2007.

[37] Gregory A Ehlers, Robert D Howerton, and Gary E Speegle. Engery management and building automation system, November 5 1996. US Patent 5,572,438.

[38] Melike Erol-Kantarci and Hussein T Mouftah. Wireless sensor networks for cost-efficient residential energy management in the smart grid. *Smart Grid, IEEE Transactions on*, 2(2) :314–325, 2011.

[39] Hossein Falaki, Ratul Mahajan, Srikanth Kandula, Dimitrios Lymberopoulos, Ramesh Govindan, and Deborah Estrin. Diversity in smartphone usage. In *Proceedings of the 8th international conference on Mobile systems, applications, and services*, pages 179–194. ACM, 2010.

[40] Stuart I Feldman. Make—a program for maintaining computer programs. *Software : Practice and experience*, 9(4) :255–265, 1979.

[41] Konstantinos P Ferentinos and Theodore A Tsiligiridis. Adaptive design optimization of wireless sensor networks using genetic algorithms. *Computer Networks*, 51(4) :1031–1051, 2007.

[42] Roy Fielding and J Reschke. Rfc 7234-hypertext transfer protocol (http/1.1) : Caching. *URl : http ://tools. ietf. org/html/rfc7234 (v isited on 02/19/2015)*.

[43] Sir Ronald Aylmer Fisher, Statistiker Genetiker, Ronald Aylmer Fisher, Statistician Genetician, Ronald Aylmer Fisher, and Statisticien Généticien. *The design of experiments*, volume 12. Oliver and Boyd Edinburgh, 1960.

[44] Eric Fleury, Nathalie Mitton, Thomas Noel, Cédric Adjih, Valeria Loscri, Anna Maria Vegni, Riccardo Petrolo, Valeria Loscri, Nathalie Mitton, Gianluca Aloi, et al. Fit iot-lab : The largest iot open experimental testbed. *ERCIM News*, (101) :14, 2015.

[45] Jeff Forcier. Fabric pythonic remote execution. http://www.fabfile.org.

[46] Travis CI GmbH. Travis CI continuous integration and deployment that just works. https://travis-ci.com/.

[47] Pietro Gonizzi, Paolo Medagliani, Giorgio Ferrari, and Jeremie Leguay. Rawmac : A routing aware wave-based mac protocol for wsns. In *Wireless and Mobile Computing, Networking and Communications (WiMob), 2014 IEEE 10th International Conference on*, pages 205–212. IEEE, 2014.

[48] David Gourley and Brian Totty. *HTTP : the definitive guide*. " O'Reilly Media, Inc.", 2002.

[49] James Grenning. Applying test driven development to embedded software. *Instrumentation & Measurement Magazine, IEEE*, 10(6) :20–25, 2007.

[50] Jayavardhana Gubbi, Rajkumar Buyya, Slaven Marusic, and Marimuthu Palaniswami. Internet of things (iot) : A vision, architectural elements, and future directions. *Future Generation Computer Systems*, 29(7) :1645–1660, 2013.

[51] Dominique Guinard, Vlad Trifa, and Erik Wilde. A resource oriented architecture for the web of things. In *Internet of Things (IOT), 2010*, pages 1–8. IEEE, 2010.

[52] Vehbi C Gungor, Bin Lu, and Gerhard P Hancke. Opportunities and challenges of wireless sensor networks in smart grid. *Industrial Electronics, IEEE Transactions on*, 57(10) :3557–3564, 2010.

[53] Jane K Hart and Kirk Martinez. Environmental sensor networks : A revolution in the earth system science ? *Earth-Science Reviews*, 78(3) :177–191, 2006.

[54] Jason Hill, Mike Horton, Ralph Kling, and Lakshman Krishnamurthy. The platforms enabling wireless sensor networks. *Communications of the ACM*, 47(6) :41–46, 2004.

[55] Robert G Hollands. Will the real smart city please stand up ? intelligent, progressive or entrepreneurial ? *City*, 12(3) :303–320, 2008.

[56] Peng Hu, Zude Zhou, Quan Liu, and Fangmin Li. The hmm-based modeling for the energy level prediction in wireless sensor networks. In *IEEE ICIEA*, Harbin, China, 2007.

[57] Urs Hunkeler, Hong Linh Truong, and Andy Stanford-Clark. Mqtt-s—a publish/subscribe protocol for wireless sensor networks. In *Communication systems software and middleware and workshops, 2008. comsware 2008. 3rd international conference on*, pages 791–798. IEEE, 2008.

[58] Damien B Jourdan and Olivier L de Weck. Multi-objective genetic algorithm for the automated planning of a wireless sensor network to monitor a critical facility. In *Defense and Security*, pages 565–575. International Society for Optics and Photonics, 2004.

[59] Simon Kellner, Mario Pink, Detlev Meier, and E-O Blass. Towards a realistic energy model for wireless sensor networks. In *Wireless on Demand Network Systems and Services, 2008. WONS 2008. Fifth Annual Conference on*, pages 97–100. IEEE, 2008.

[60] Branko Kerkez, Steven D Glaser, Roger C Bales, and Matthew W Meadows. Design and performance of a wireless sensor network for catchment-scale snow and soil moisture measurements. *Water Resources Research*, 48(9), 2012.

[61] Kavi K Khedo, Rajiv Perseedoss, Avinash Mungur, et al. A wireless sensor network air pollution monitoring system. *arXiv preprint arXiv :1005.1737*, 2010.

[62] Sukun Kim, Shamim Pakzad, David Culler, James Demmel, Gregory Fenves, Steven Glaser, and Martin Turon. Health monitoring of civil infrastructures using wireless sensor networks. In *Information processing in sensor networks, 2007. IPSN 2007. 6th international symposium on*, pages 254–263. IEEE, 2007.

[63] Jonathan G Koomey, Stephen Berard, Marla Sanchez, and Henry Wong. Implications of historical trends in the electrical efficiency of computing. *Annals of the History of Computing, IEEE*, 33(3) :46–54, 2011.

[64] Matthias Kovatsch, Simon Duquennoy, and Adam Dunkels. A low-power coap for contiki. In *Mobile Adhoc and Sensor Systems (MASS), 2011 IEEE 8th International Conference on*, pages 855–860. IEEE, 2011.

[65] Matthias Kovatsch, Martin Lanter, and Zach Shelby. Californium : Scalable cloud services for the internet of things with coap. In *Internet of Things (IOT), 2014 International Conference on the*, pages 1–6. IEEE, 2014.

[66] N. Kushalnagar, G. Montenegro, and C. Schumacher. IPv6 over Low-Power Wireless Personal Area Networks (6LoWPANs) : Overview, Assumptions, Problem Statement, and Goals. RFC 4919 (Informational), August 2007.

[67] Mathieu Lacage, Martin Ferrari, Mads Hansen, Thierry Turletti, and Walid Dabbous. Nepi : using independent simulators, emulators, and testbeds for easy experimentation. *ACM SIGOPS Operating Systems Review*, 43(4) :60–65, 2010.

[68] Abdelkader Lahmadi, Alexandre Boeglin, and Olivier Festor. Efficient distributed monitoring in 6lowpan networks. In *CNSM*, Zurich, Switzerland, 2013.

[69] Koen Langendoen. Medium access control in wireless sensor networks. *Medium access control in wireless networks*, 2 :535–560, 2008.

[70] Jung Hoon Lee, Robert Phaal, and Sang-Ho Lee. An integrated service-device-technology roadmap for smart city development. *Technological Forecasting and Social Change*, 80(2) :286–306, 2013.

[71] Tomas Lennvall, Stefan Svensson, and Fredrik Hekland. A comparison of wirelesshart and zigbee for industrial applications. In *IEEE International Workshop on Factory Communication Systems*, volume 2008, pages 85–88, 2008.

[72] Rémy Léone, Jérémie Leguay, Paolo Medagliani, and Claude Chaudet. Demo abstract : Makesense—managing reproducible wsns experiments. In *Real-World Wireless Sensor Networks*, pages 65–71. Springer, 2014.

[73] Rémy Leone, Jeremie Leguay, Paolo Medagliani, Claude Chaudet, et al. Makesense : Managing reproducible wsns experiments. *Fifth Workshop on Real-World Wireless Sensor Networks*, 2013.

[74] Slawek Ligus. *Effective Monitoring and Alerting*. " O'Reilly Media, Inc.", 2012.

[75] Changlei Liu and Guohong Cao. Distributed monitoring and aggregation in wireless sensor networks. In *IEEE INFOCOM*, San Diego, CA, USA, 2010.

[76] Keqin Liu and Qing Zhao. Intrusion detection in resource-constrained cyber networks : A restless multi-armed bandit approach. *submitted to IEEE/ACM Transactions on Networking. Avialable at http ://arxiv. org/abs*, 1112.

[77] David G Loomis and Lester D Taylor. *Forecasting the Internet : understanding the explosive growth of data communications*, volume 39. Springer Science & Business Media, 2012.

[78] Sean Luke. *Essentials of Metaheuristics*. Lulu, second edition, 2013. Available for free at http ://cs.gmu.edu/∼sean/book/metaheuristics/.

[79] Alan Mainwaring, David Culler, Joseph Polastre, Robert Szewczyk, and John Anderson. Wireless sensor networks for habitat monitoring. In *Proceedings of the 1st ACM international workshop on Wireless sensor networks and applications*, pages 88–97. ACM, 2002.

[80] Jerry Martocci, Pieter Mil, Nicolas Riou, and Wouter Vermeylen. Building automation routing requirements in low-power and lossy networks. 2010.

[81] Wes McKinney. Data structures for statistical computing in python. In Stéfan van der Walt and Jarrod Millman, editors, *Proceedings of the 9th Python in Science Conference*, pages 51 – 56, 2010.

[82] Paolo Medagliani, Jérémie Leguay, Andrzej Duda, Franck Rousseau, Marc Domingo, Mischa Dohler, Ignasi Vilajosana, and Olivier Dupont. Bringing ip to low-power smart objects : the smart parking case in the calipso project.

[83] Scott Michel, Khoi Nguyen, Adam Rosenstein, Lixia Zhang, Sally Floyd, and Van Jacobson. Adaptive web caching : towards a new global caching architecture. *Computer Networks and ISDN systems*, 30(22) :2169–2177, 1998.

[84] Raquel A.F. Mini, Antonio A.F. Loureiro, and Badri Nath. The distinctive design characteristic of a wireless sensor network : the energy map. *Computer Communications*, 27, 2004.

[85] Javier Moreno Molina, Jan Haase, and Christoph Grimm. Energy consumption estimation and profiling in wireless sensor networks. In *Architecture of Computing Systems (ARCS), 2010 23rd International Conference on*, pages 1–6. VDE, 2010.

[86] G. Montenegro, N. Kushalnagar, J. Hui, and D. Culler. Transmission of IPv6 Packets over IEEE 802.15.4 Networks. RFC 4944 (Proposed Standard), September 2007. Updated by RFC 6282.

[87] David Moss and Philip Levis. Box-macs : Exploiting physical and link layer boundaries in low-power networking. *Computer Systems Laboratory Stanford University*, pages 116–119, 2008.

[88] San Murugesan. Harnessing green it : Principles and practices. *IT professional*, 10(1) :24–33, 2008.

[89] B O'Flyrm, Ricardo Martinez, John Cleary, Catherine Slater, F Regan, Dermot Diamond, and H Murphy. Smartcoast : a wireless sensor network for water quality monitoring. In *Local Computer Networks, 2007. LCN 2007. 32nd IEEE Conference on*, pages 815–816. Ieee, 2007.

[90] Fredrik Österlind, Adam Dunkels, Joakim Eriksson, Niclas Finne, and Thiemo Voigt. Cross-level sensor network simulation with cooja. In *LCN*, 2006.

[91] Tae Rim Park, Tae Hyun Kim, Jae Young Choi, Sunghyun Choi, and Wook Hyun Kwon. Throughput and energy consumption analysis of ieee 802.15. 4 slotted csma/ca. *Electronics Letters*, 41(18) :1017–1019, 2005.

[92] Al-Sakib Khan Pathan, Hyung-Woo Lee, and Choong Seon Hong. Security in wireless sensor networks : issues and challenges. In *Advanced Communication Technology, 2006. ICACT 2006. The 8th International Conference*, volume 2, pages 6–pp. IEEE, 2006.

[93] Roger D Peng. Reproducible research in computational science. *Science (New York, Ny)*, 334(6060) :1226, 2011.

[94] Fernando Pérez and Brian E. Granger. IPython : a system for interactive scientific computing. *Computing in Science and Engineering*, 9(3) :21–29, May 2007.

[95] Raphael Pham, Leif Singer, Olga Liskin, Fernando Figueira Filho, and Klaus Schneider. Creating a shared understanding of testing culture on a social coding site. In *Software Engineering (ICSE), 2013 35th International Conference on*, pages 112–121. IEEE, 2013.

[96] J. Polastre, R. Szewczyk, and D. Culler. Telos : enabling ultra-low power wireless research. In *Proc. of the 4th Int. Symp. on Information Processing in Sensor Networks (*IPSN 05), pages 364 – 369, Piscataway, NJ, 2005.

[97] Karl Raimund Popper. *The Logic of Scientific Discovery*. Routledge, 2002. 1st English Edition :1959.

[98] Narenda PrithviRaj, Simon Duquennoy, and Thiemo Voigt. Ble and ieee 802.15. 4 in the iot : Evaluation and interoperability considerations. In *International Conference on Interoperability in IoT*, 2015.

[99] Daniele Puccinelli and Martin Haenggi. Multipath fading in wireless sensor networks : measurements and interpretation. In *Proceedings of the 2006 international conference on Wireless communications and mobile computing*, pages 1039–1044. ACM, 2006.

[100] Guy Pujolle. *Les réseaux : Edition 2014*. Editions Eyrolles, 2014.

[101] Jeff Racine. gnuplot 4.0 : a portable interactive plotting utility. *Journal of Applied Econometrics*, 21(1) :133–141, 2006.

[102] Will Reese. Nginx : the high-performance web server and reverse proxy. *Linux Journal*, 2008(173) :2, 2008.

[103] Leonard Richardson and Sam Ruby. *RESTful web services*. " O'Reilly Media, Inc.", 2008.

[104] Luis Ruiz-Garcia, Loredana Lunadei, Pilar Barreiro, and Ignacio Robla. A review of wireless sensor technologies and applications in agriculture and food industry : state of the art and current trends. *Sensors*, 9(6) :4728–4750, 2009.

[105] Z. Shelby. Constrained RESTful Environments (CoRE) Link Format. RFC 6690 (Proposed Standard), August 2012.

[106] Z. Shelby, K. Hartke, and C. Bormann. The Constrained Application Protocol (CoAP). RFC 7252 (Proposed Standard), June 2014.

[107] Zach Shelby and Carsten Bormann. *6LoWPAN : The wireless embedded Internet*, volume 43. John Wiley & Sons, 2011.

[108] Helen Shen et al. Interactive notebooks : Sharing the code. *Nature*, 515(7525) :151–152, 2014.

[109] Silvair. Bluetooth : A technology in transition. `https://blog.silvair.com/2015/11/26/bluetooth-a-technology-in-transition/`, 2015.

[110] Silvair. Riding the z-wave. `https://blog.silvair.com/2015/10/15/wireless-protocols-showdown-riding-the-z-wave/`, 2015.

[111] Silvair. Threading the way through a connected home. `https://blog.silvair.com/2015/11/12/wireless-protocols-showdown-threading-the-way-through-a-connected-home/`, 2015.

[112] Silvair. Wireless protocols showdown : Why not wi-fi ? `https://blog.silvair.com/2015/10/01/wireless-protocols-showdown-3/`, 2015.

[113] Silvair. Wireless protocols showdown : Zigbee – is the sting still sharp ? `https://blog.silvair.com/2015/10/27/zigbee-is-the-sting-still-sharp/`, 2015.

[114] John Ferguson Smart. *Jenkins : the definitive guide*. " O'Reilly Media, Inc.", 2011.

[115] Zhenyu Song, Mihai T Lazarescu, Riccardo Tomasi, Luciano Lavagno, and Maurizio A Spirito. High-level internet of things applications development using wireless sensor networks. In *Internet of Things*, pages 75–109. Springer, 2014.

[116] Thanos Stathopoulos, John Heidemann, and Deborah Estrin. A remote code update mechanism for wireless sensor networks. Technical report, DTIC Document, 2003.

[117] Lu Tan and Neng Wang. Future internet : The internet of things. In *Advanced Computer Theory and Engineering (ICACTE), 2010 3rd International Conference on*, volume 5, pages V5–376. IEEE, 2010.

[118] R Core Team. R : A language and environment for statistical computing. r foundation for statistical computing, vienna, austria. 2013, 2014.

[119] Andreas Terzis, Annalingam Anandarajah, Kevin Moore, I Wang, et al. Slip surface localization in wireless sensor networks for landslide prediction. In *Proceedings of the 5th international conference on Information processing in sensor networks*, pages 109–116. ACM, 2006.

[120] Joydeep Tripathi, Jaudelice Cavalcante de Oliveira, and Jean-Philippe Vasseur. A performance evaluation study of rpl : Routing protocol for low power and lossy networks. In *Information Sciences and Systems (CISS), 2010 44th Annual Conference on*, pages 1–6. IEEE, 2010.

[121] Nicolas Tsiftes, Joakim Eriksson, and Adam Dunkels. Low-power wireless ipv6 routing with contikirpl. In *Proceedings of the 9th ACM/IEEE International Conference on Information Processing in Sensor Networks*, pages 406–407. ACM, 2010.

[122] Hans van der Veer and Anthony Wiles. Achieving technical interoperability. *European Telecommunications Standards Institute*, 2008.

[123] Lorenzo Vangelista, Andrea Zanella, and Michele Zorzi. Long-range iot technologies : The dawn of lora™. In *Future Access Enablers for Ubiquitous and Intelligent Infrastructures*, pages 51–58. Springer, 2015.

[124] Malisa Vucinic, Bernard Tourancheau, and Andrzej Duda. Performance Comparison of the RPL and LOADng Routing Protocols in a Home Automation Scenario. In *IEEE WCNC*, 2013.

[125] Klaus-Dieter Walter. Implementing m2m applications via gprs, edge and umts. *White paper—M2M Alliance*, 2009.

[126] Ning Wang, Naiqian Zhang, and Maohua Wang. Wireless sensors in agriculture and food industry—recent development and future perspective. *Computers and electronics in agriculture*, 50(1) :1–14, 2006.

[127] Yong Wang, Garhan Attebury, and Byrav Ramamurthy. A survey of security issues in wireless sensor networks. 2006.

[128] Tim Wark, Peter Corke, Pavan Sikka, Lasse Klingbeil, Ying Guo, Chris Crossman, Phil Valencia, Dave Swain, and Greg Bishop-Hurley. Transforming agriculture through pervasive wireless sensor networks. *Pervasive Computing, IEEE*, 6(2) :50–57, 2007.

[129] Geoffrey Werner-Allen, Konrad Lorincz, Mario Ruiz, Omar Marcillo, Jeff Johnson, Jonathan Lees, and Matt Welsh. Deploying a wireless sensor network on an active volcano. *Internet Computing, IEEE*, 10(2) :18–25, 2006.

[130] Duane Wessels. *Web caching*. " O'Reilly Media, Inc.", 2001.

[131] Thomas Williams and Lars Hecking. Gnuplot. 2003.

[132] Thomas Williams, Colin Kelley, and many others. Gnuplot 4.4 : an interactive plotting program. `http://gnuplot.info/`, March 2010.

[133] Greg Wilson, DA Aruliah, C Titus Brown, Neil P Chue Hong, Matt Davis, Richard T Guy, Steven HD Haddock, Katy Huff, Ian M Mitchell, Mark D Plumbley, et al. Best practices for scientific computing. *PLoS biology*, 12(1) :e1001745, 2014.

[134] T. Winter, P. Thubert, A. Brandt, J. Hui, R. Kelsey, P. Levis, K. Pister, R. Struik, JP. Vasseur, and R. Alexander. RPL : IPv6 Routing Protocol for Low-Power and Lossy Networks. RFC 6550 (Proposed Standard), March 2012.

[135] T. Winter, P. Thubert, A. Brandt, J. Hui, R. Kelsey, P. Levis, K. Pister, R. Struik, JP. Vasseur, and R. Alexander. RPL : IPv6 Routing Protocol for Low-Power and Lossy Networks. RFC 6550 (Proposed Standard), March 2012.

[136] Kehui Xiao, Deqin Xiao, and Xiwen Luo. Smart water-saving irrigation system in precision agriculture based on wireless sensor network. *Transactions of the Chinese Society of Agricultural Engineering*, 26(11) :170–175, 2010.

[137] Ye Yan, Yi Qian, Hamid Sharif, and David Tipper. A survey on smart grid communication infrastructures : Motivations, requirements and challenges. *Communications Surveys & Tutorials, IEEE*, 15(1) :5–20, 2013.

[138] Jocelyn Young. Science interactive notebooks in the classroom. *Science Scope*, 26(4) :44–57, 2003.

[139] Liyang Yu, Neng Wang, and Xiaoqiao Meng. Real-time forest fire detection with wireless sensor networks. In *Wireless Communications, Networking and Mobile Computing, 2005. Proceedings. 2005 International Conference on*, volume 2, pages 1214–1217. IEEE, 2005.

[140] Jerry Zhao, Ramesh Govindan, and Deborah Estrin. Residual energy scans for monitoring wireless sensor networks. In *IEEE WCNC*, 2002.

Passerelle intelligente pour réseaux de capteurs

Remy Leone

RESUME :

MOTS-CLEFS : bla bla bla

ABSTRACT :

KEY-WORDS : bla bla bla