Rapport projet réseau sans-fils

Nathanaël Derousseaux

Vagnona Andrianandrasana-Dina

https://git.unistra.fr/vandrianandrasan/reseau-sans-fil

I - Variables étudiées

Nos étudieront l'impact de certaines variables sur les performances du protocole TSCH et de l'ordonnancement Orchestra. Ces variables seront les suivantes :

- Nombre de senders
- Taille d'un slot

A - Nombre de senders

Faire varier le nombre de senders est peut-être la variable la plus simple, mais la plus intéressante à étudier. En effet, comme pour tout protocole, il est important qu'il puisse tenir la montée en charge.

B - Taille d'un slot

Au cours de nos expériences, nous allons faire varier la taille des slots. En faisant varier la taille des slots, nous faisons varier la taille de la supertrame. Nous espérons ainsi observer une variation sur le délai ou les débits utiles avec TSCH.

C - Comparaison avec d'autres protocoles MAC

Comparer le protocole TSCH avec d'autres protocoles MAC aurait été très intéressant, cependant il n'a pas pu aboutir durant notre travail par manque de temps.
Nous aurions pu comparer les différentes métriques citées par la suite avec le protocole TSCH et révéler qu'elles sont les avantages et inconvénients du protocole TSCH par rapport aux autres protocoles MAC.

II - Définition des métriques mesurées

Au cours de nos expérimentations, nous allons nous pencher sur plusieurs métriques :

- La consommation énergétique
- Le débit utile
- Taux de pertes
- Délai

Vous pouvez retrouver nos scripts utilisés pour agréger et traiter les données dans le dossier `Plots` de notre git. Les classes Scenario et Node offrent une boîte à outils pour générer facilement des résumés des données.
Par exemple:

```
s = Scenario(”Scenarios/s0-etalon”, 5, TSCH, 397, 600)
print(s)
```

Affiche un résumé des valeurs intéressantes du scénario. Les graphiques seront générés par mathplotlib.

A - Consommation énergétique

Si la consommation énergétique est un facteur négligeable pour les réseaux filaires, il n’en est pas de même avec les réseaux sans-fils. En effet les clients sont des appareils portables et donc souvent sur batterie.
Au cours de nos scénarios nous allons donc pouvoir comparer l’évolution de la consommation moyenne en fonction des variables que nous allons ajuster.
Nos scripts permettent de calculer la consommation moyenne en Wh, mais on préférera utiliser la moyenne de la puissance instantanée (en W), plus générale car elle ne dépend pas de la durée de l’expérience.
On comparera principalement la consommation des senders, car dans un cas réel, le coordinateur est rarement sur batterie. Seule la consommation des senders est donc pertinente.

B - Débit utile

La consommation énergétique est peut-être importante, mais la métrique fondamentale des réseaux est le débit utile. En effet, on attend une certaine capacité de download et d’upload de la part de nos réseaux wifi.
Nos scripts permettent de mesurer la moyenne du débit utile de tous les senders. Il n’est pas nécessaire d’ajouter celui du coordinateur, car comme il répond a chaque trame UDP qu’il reçoit, cela fausserait nos données. Du plus on s'intéresse à un cas réel, ou un utilisateur voudrait utiliser le réseau depuis son téléphone portable par exemple. Ce qui nous intéresse ici, c’est de connaître le débit que peut espérer cet utilisateur.

C - Taux de perte

Dans les réseaux sans fil, la fiabilité de la communication est également très importante. Les pertes de paquets peuvent causer des retards et des ré-transmissions qui peuvent ralentir le réseau et entraîner des problèmes de qualité de service. Des pertes de paquets importantes peuvent également entraîner une diminution du débit utile et une augmentation de la consommation d'énergie due aux ré-transmissions.

D - Délai

Le délai correspond ici, au temps (du point de vue de l'expéditeur) qu'à mis un paquet pour être envoyé par l'expéditeur vers le coordinateur jusqu'au moment où l'on reçoit la réponse du coordinateur. Cela permet d'avoir une bonne vision de comment le coordinateur gère la charge qu'il reçoit. En effet, logiquement plus on ajoute des nœuds plus le délai devrait augmenter mais la question que nous nous posons est : comment augmente-t-elle ?

III - Scénarios

Pour mesurer l'impact de nos variables sur les métriques que nous avons décidés d'étudier nous allons faire jouer un certain nombre de scénarios, chaque scénario isolant une variable.

Ainsi, nous aurons un scénario étalon, nommé scénario 0, permettant de comparer toutes nos modifications. Ce scénario possède 5 senders et une taille de slot de 397 microsecondes.

Un premier scénario testera la montée en charge en augmentant le nombre de senders. Il y aura 3 expériences : 1 nœud, 2 nœuds et 10 nœuds. Le scénario étalon, possédant 5 nœuds on aura des données pour des valeurs allant de 1 à 10 nœuds. Nous aurions réellement voulu augmenter le nombre de nœuds, mais comme les nœuds sur la plateforme étaient souvent pris, et que nos camarades en avaient besoin autant que nous, nous avons décidé de ne pas monopoliser trop de nœuds. Il était surtout difficile de trouver des noeuds de libre. Pour la même raison, nous avons fait durer chaque expérience 10 minutes.

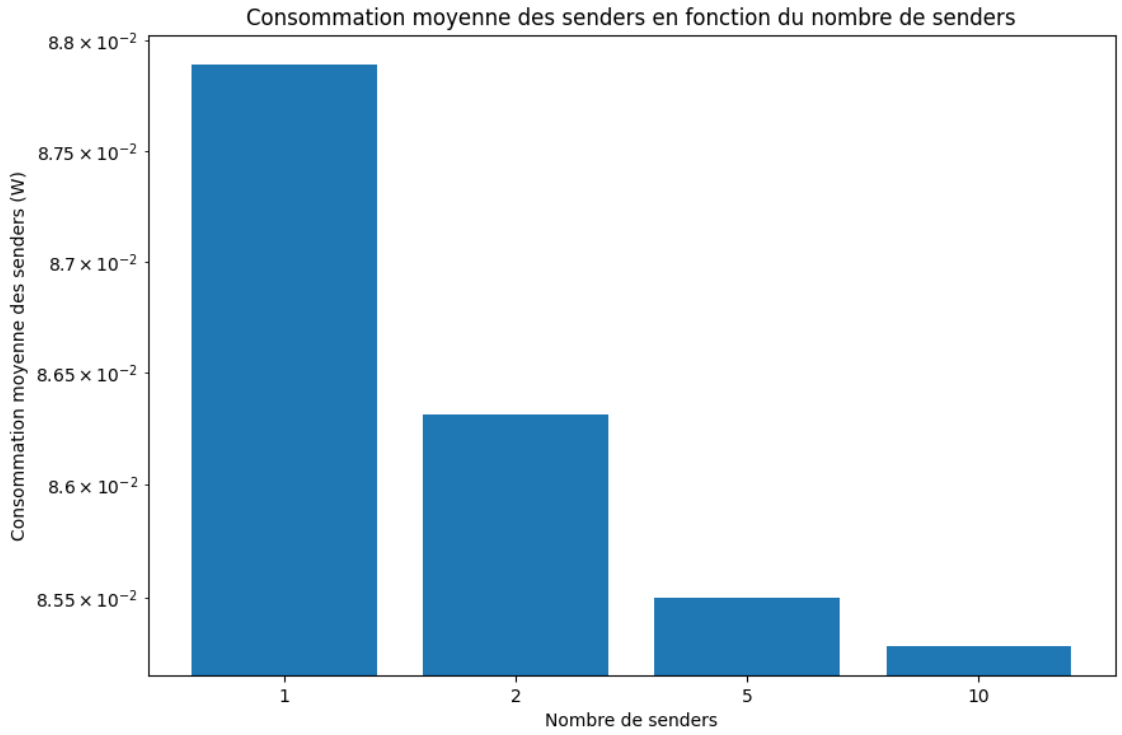
Un second scénario testera l'impact de la taille des slots sur les performances de TSCH. Cette taille de slot prendra les valeurs 100, 200 et 1000 microsecondes. Le scénario étalon utilisant un slot de taille 397, on aura des tailles de slot allant de 100 à 1000. Étant donné qu'il y a 16 slot par supertrame, alors la taille de la supertrame variera entre 1600, et 16000 microsecondes.

Nous avons fait tourner plusieurs fois chaque expérience afin de pouvoir faire une moyenne et d'augmenter la précision de nos données.
Les scripts permettant de jouer ces scénarios de manière automatique peuvent être trouvés dans le dossier `Scenario` de notre git.

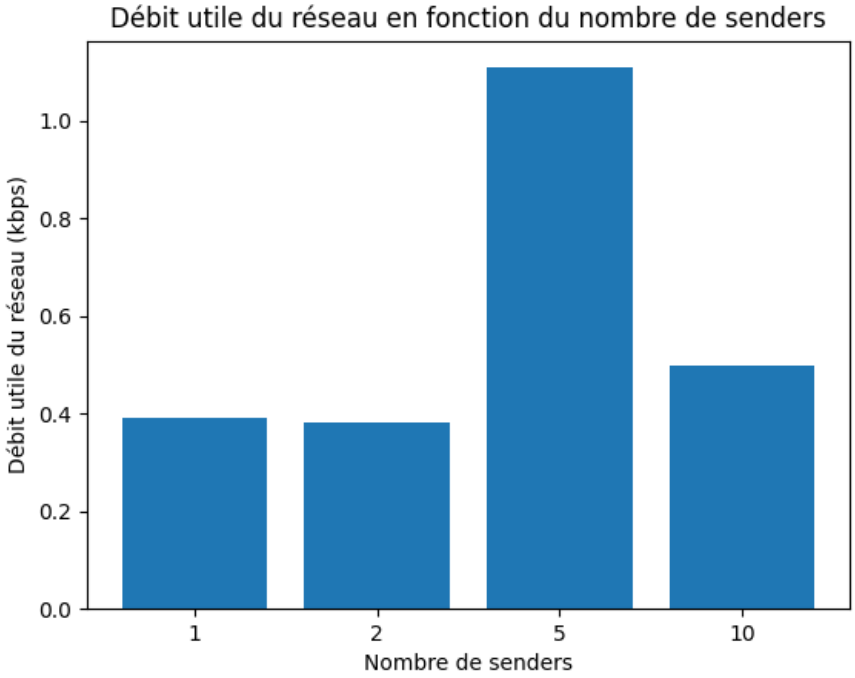
IV - Résultats

A - Impact du nombre de noeud

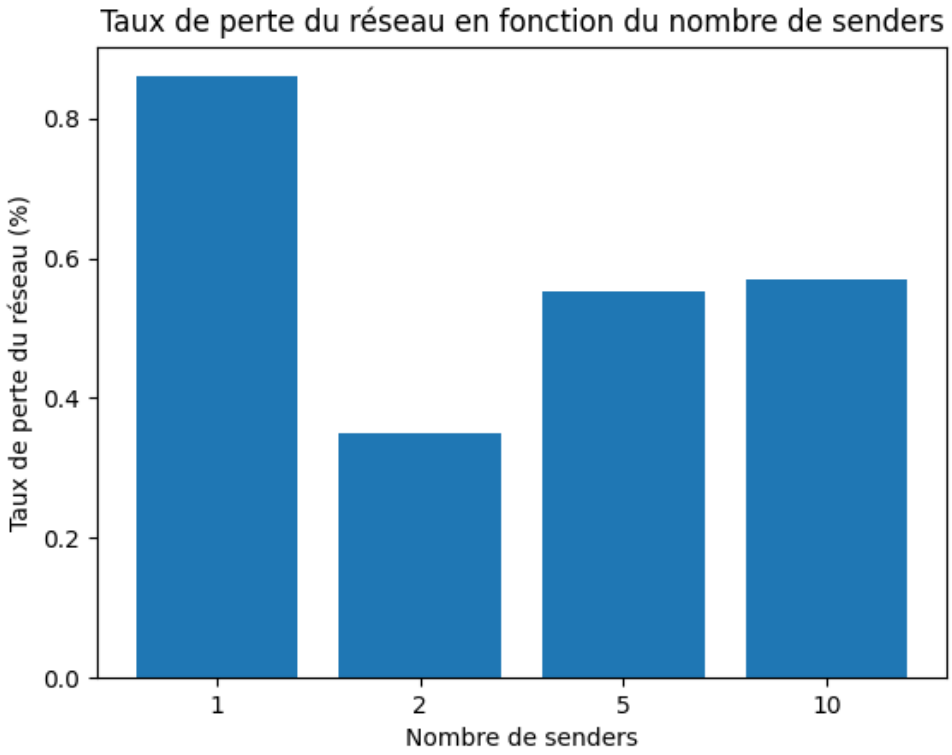
Pour ce qui est de la consommation énergétique, on peut voir, sur la figure ci-dessous, que celle çi reste sensiblement la même en fonction du nombre de nœuds.



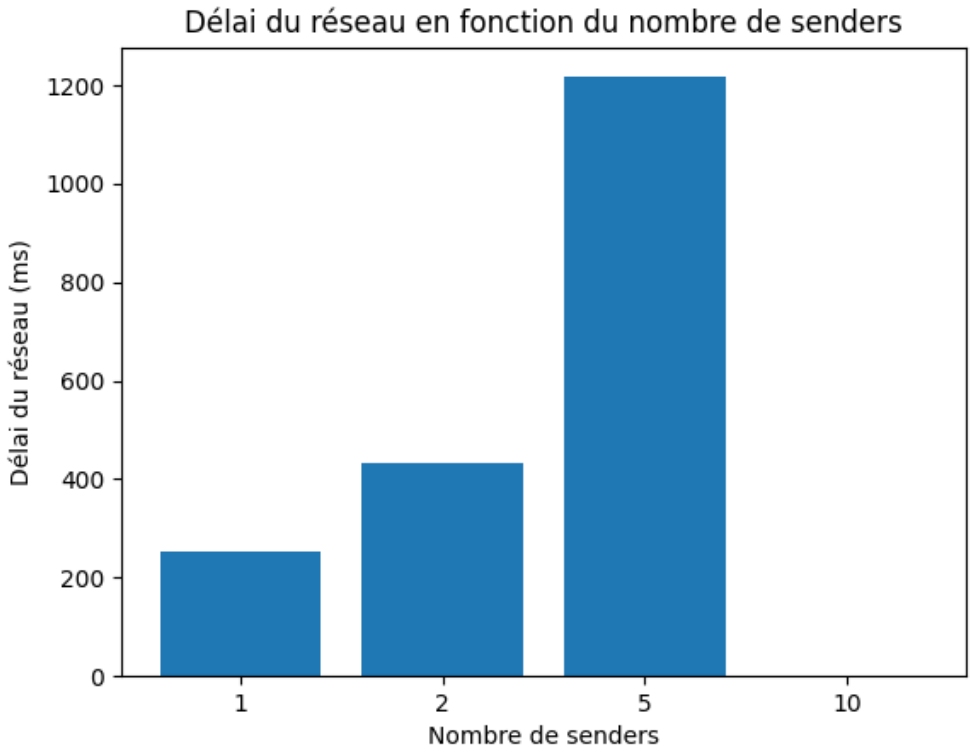
Malgré tout, avec l'échelle logarithmique, on constate que la consommation décroît à mesure que le nombre de nœuds augmente. C'est là aussi assez logique, plus il y a de nœuds sur le réseau, plus il y a de délai et plus le temps passé à attendre pour les senders (et donc pas à émettre) augmente. La consommation moyenne décroît donc à mesure que le nombre de senders augmente.
Sur la figure suivante, on peut voir que le débit utile moyen est à son maximum quand le nombre de senders est de 5. Ce résultat est étonnant, car le maximum devrait être pour 1 sender. En effet, si il n'y a qu'un seul sender, on a besoin de beaucoup moins de mécanismes de synchronisation, et du coup presque tout le débit est utilisé à des fins utiles. Peut-être le problème vient-il d'une erreur de notre part dans la manière que l'on a de mesurer le débit.



Sur la figure suivante, on constate là encore une certaine incohérence sur le taux de pertes. En effet, le taux de perte devrait être minimum avec un seul sender (moins d'événements de contention). Peut-être encore une erreur dans la réalisation de nos expériences.

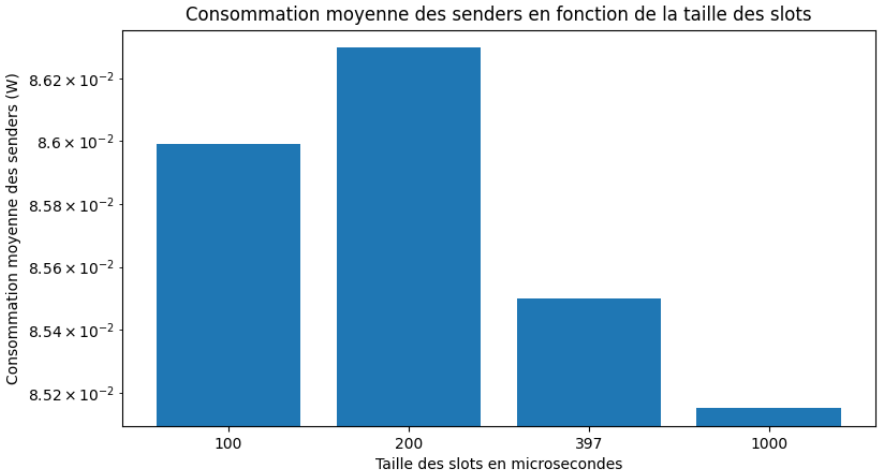


Sur ce dernier schéma, on constate l'intuition que l'on avait sur le fait que le nombre de senders augmente. En effet, avoir plus de senders demande de partager le réseau, et donc individuellement chaque senders attend un peu plus longtemps avant de recevoir sa réponse.

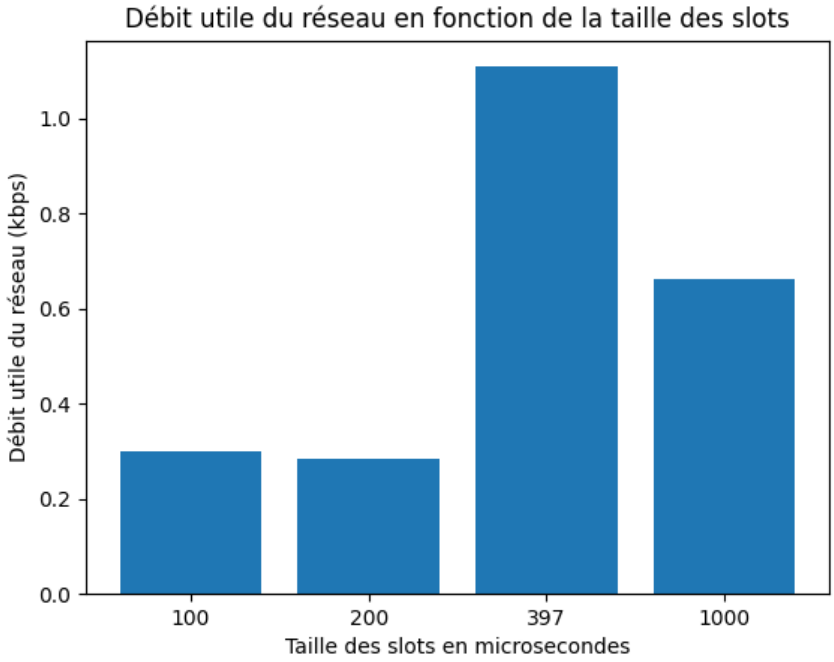


Nous n'avons pas pu mesurer le délai avec 10 senders à cause du manque de nœuds disponible à la fin du projet.

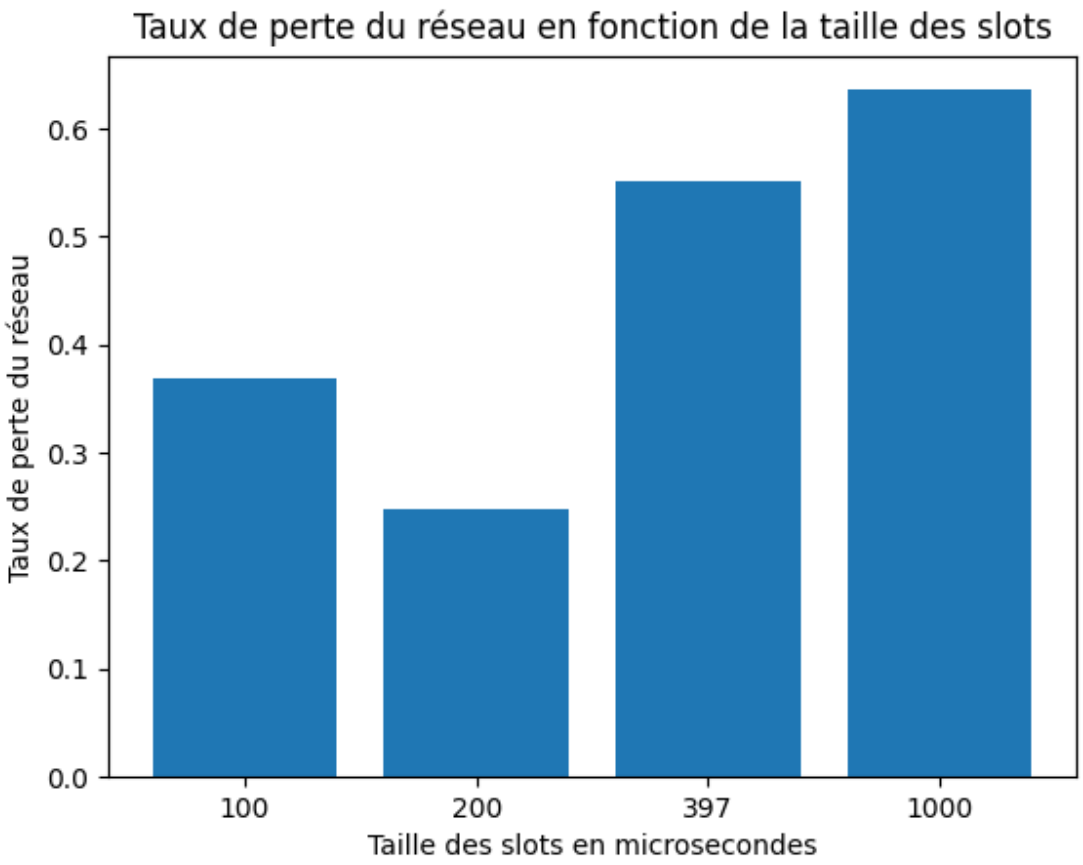
B - Impact de taille des slots

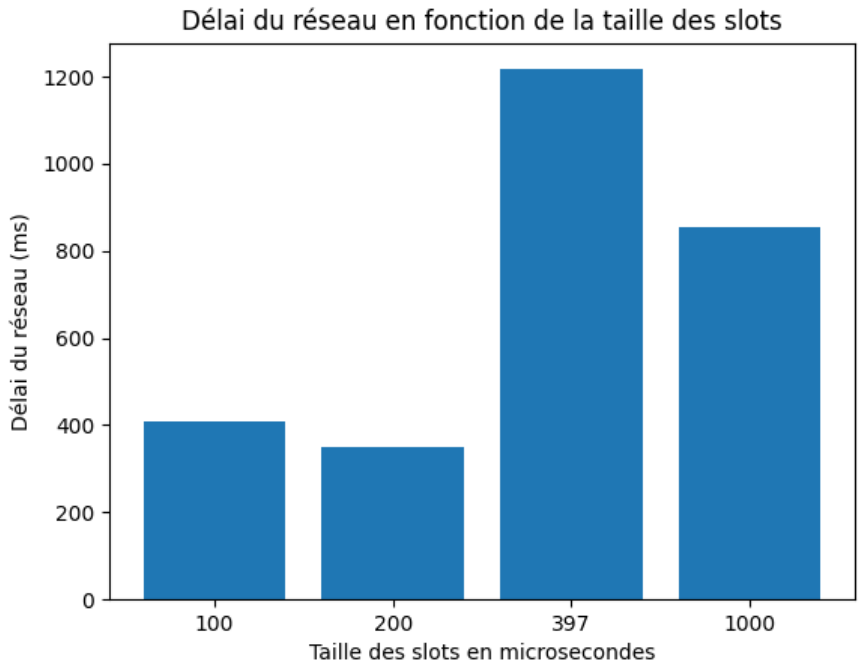


Ici on constate que 200 à l'air d'être le pic de consommation. Ensuite, plus la taille des slots augmente, plus la consommation diminue. C'est assez attendu car si les senders envoient une trame plus grande, il sera moins souvent nécessaire de changer de mode.
On peut nuancer cependant en disant que les variations dans la consommation sont très faibles, et sans doute négligeables dans un cas réel.
Sur la figure ci-dessus on constate que 397 à l'air d'être l'optimal en termes de taille de slot. On peut supposer que c'est le parfait juste milieu entre grosses quantité de données envoyées par slot, et un bon partage du réseau.



Sur la figure ci-dessus, on peut voir le taux de pertes en fonction de la taille des slots. Encore une fois, on constate qu'une valeur moyenne est optimale. On peut supposer que les mécanismes de synchronisation sont plus efficaces et qu'on observe dans ce cas-là, moins d'événements de contentions.





Sur le dernier schéma on peut observer que 397 microsecondes est le pire cas pour le délai. Le débit utile étant plus grand pour cette valeur, cela n'est sans doute pas un problème. En effet, il est souvent préférable d'avoir un débit utile plus grand au détriment du délai. 397 microsecondes est sans doute un bon compromis.

Pour conclure, il faudrait avoir positionné la taille des slots entre 200 ms et 400 ms afin de minimiser le taux de perte et/ou le débit utile.
Cependant, on constate une contradiction avec la mesure du délai, puisque pour une taille de slot à 397 on a le délai le plus grand alors que le débit utile est le plus grand également.