Couche de Liaison – scripts de transmission

Objectifs du laboratoire
Appliquer les connaissances acquises dans la partie théorique à la conception et implémentation d'une couche de liaison.

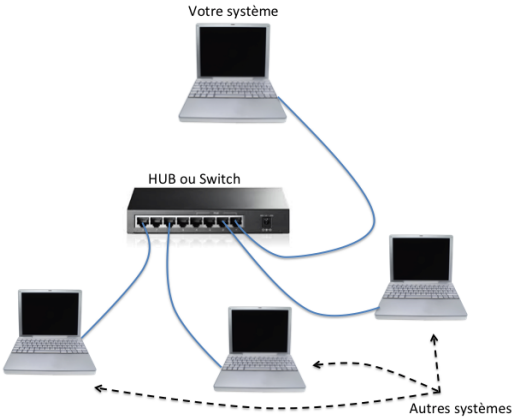

Fig. 1. Réseau sur lequel votre couche de Liaison doit fonctionner.

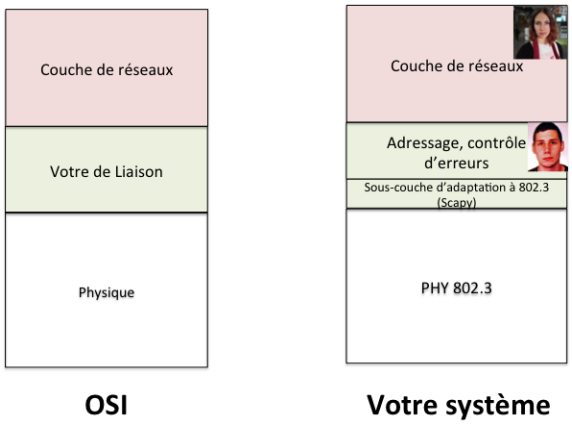

Fig. 2. Architecture en couches du protocole. A gauche, modèle OSI. A droite, modèle correspondant à votre système.

Vous allez utiliser les services de la couche physique de 802.3 (Ethernet) pour transmettre vos trames. Pour que les trames soient envoyées par cette couche physique, il faudra les encapsuler dans une trame MAC 802.3. Cette encapsulation est la responsabilité de la sous-couche d'adaptation montrée à la figure 2.

La sous-couche d'adaptation doit encapsuler les trames de votre protocole dans des trames MAC 802.3 ayant comme adresse de destination l'adresse broadcast 802.3 (ff:ff:ff:ff:ff:ff). Vous pouvez définir le type de la trame. Le type dans les trames Ethernet se trouve dans les deux octets qui suivent l'adresse de destination MAC.

Une fois la couche de Liaison conçue et implémentée, le réseau doit fonctionner comme suit :

1. La couche de réseaux, exécutée par le personnel enseignant, vous donnera une primitive de service contenant des données (texte à convertir en ascii 8bits) et l'identifiant du système destinataire.
Ex : data = « La raclette de Noel, c'est à quelle heure ? »
dest = 1101 (selon votre table d'adresse)
src = 0001 (selon votre table d'adresse)

2. Vous devrez alors construire une trame suivant le protocole de votre couche de Liaison.

3. La trame doit ensuite être encapsulée dans une trame MAC 802.3 et injectée au niveau de la couche physique pour transmission par le câble 802.3 vers le HUB/Switch (voir figure 1).

4.La trame sera reçue par les autres systèmes dans le réseau, où elle atteindra les entités de votre couche de Liaison. Vous devrez alors la traiter pour détecter et corriger les erreurs éventuelles. Une fois satisfait-e-s de l'intégrité de la trame, vous passerez les données à la couche de réseau dans une primitive de service.

Déroulement
Ce laboratoire est la suite du laboratoire précédent où vous avez conçu votre protocole de couche 2.
Ce projet comporte une démonstration au personnel enseignant le 26.01.2018. Des informations sur le déroulement exacte seront données ultérieurement.

Lors de la démonstration, on vous demandera d'envoyer des trames avec scapy d'un ordinateur à un autre manuellement. Ces trames contiendront les données que la couche 3 (le personnel enseignant) vous auront passées. Vous devrez donc analyser sur le moment les trames que le destinataire recevra et forger à la volée les trames qui répondent aux messages reçus en accord avec votre protocole.

Les scripts cités ci-dessous ont pour but de gagner du temps lors de la démonstration. Cependant les implémentations de ces scripts sont propres à vos protocoles.

Vous êtes libres d'ajouter d'autres scripts pour d'autres parties de votre protocole, du moment qu'il est possible d'avoir une traçabilité de ce qui est fait lors de la démonstration et qu'il soit possible pour le personnel enseignant d'ajouter volontairement des erreurs pour tester votre protocole avant d'utiliser le script SendPPDU.

Dans ce labo, vous créerez une librairie Python contenant des fonctions selon les spécification suivantes :

Nom de la fonction: CreateLPDU
But de la fonction: créer les PDUs de votre protocole de la couche de Liaison de données.

Arguments d'entrée : adresse de destination, adresse de source, cargaison, type.
Les adresses et la cargaison seront données en ascii. Par exemple, si l'adresse de destination est 0110, l'adresse de source est 1111, et la cargaison est tweet, le script prendra comme arguments le texte '0110', '1111', 'tweet' et le type de trame (données ou ACK).
Sortie : Une suite d'octets (avec correspondance ascii) représentant les bits de votre LPDU (données ou ACK).

Nom de la fonction: SendPPDU
But de la fonction: encapsuler un LPDU dans une trame Ethernet et la transmettre.

Arguments d'entrée : Une suite d'octets représentant les bits de votre LPDU. C'est la sortie de la CreateLPDU.
Sortie : Trame Ethernet avec adresse de source 00:00:00:00:00:00 et adresse destination ff:ff:ff:ff:ff:ff contenant dans son payload votre trame LPDU. La trame Ethernet doit être transmise.

On doit pouvoir capturer la trame transmise par le script 2 sur Wireshark dans la station destinatrice.

Nom de la fonction: Fragment
But de la fonction: fragmenter un NPDUs de maximum 50 octets en fragments de longueur d'au maximum 21 octets.

Arguments d'entrée : Une suite d'octets ascii représentant les bits de la NPDU. Donnée par le personnel enseignant.
Sortie : Fragments d'au plus 21 octets.

Nom de la fonction: CreateLSDU
Fonction inverse de la fonction CreateLPDU qui doit indiquer/afficher si :
- il n'y a aucune erreur
- il y avait une erreur et elle a été corrigée
- il y a entre 2 et 3 erreurs ou une raffale de longeur < 8 qui a été détectée

Sortie : le LSDU si on peut le trouver, -1 en cas d'erreurs incorrigibles.

Nom de la fonction: ReceivePPDU
But de la fonction: Filtrer les PPDU qui nous sont destinés et les stocker dans un tableau.
Fonction complémentaire du script SendPPDU.

Nom de la fonction: Defragment
Fonction complémentaire de la fonction Fragment.

Arguments d'entrée : Fragments d'au plus 21 octets faisant partie de la même séquence.
Sortie : Une suite d'octets ascii représentant les bits de la NPDU.