PROJET - 2ème année - RSA Réseaux

2019-2020

MyTFTP

Modalités

— Le PROJET sera réalisé en binôme. Il est à rendre pour le **3 mai 2020 à 23h59**.

— Les soutenances seront organisées **le mardi 5 mai et le mercredi 6 mai 2020**.

— Vous devez utiliser la plate-forme GitLab de l'école pour héberger une version de votre programme. Le projet sera privé et son identifiant sera de la forme Projet_RSA_Nom1_Nom2 (où Nom1 et Nom2 correspondent aux noms des membres du binôme). Donnez également les droits d'accès à Rémi Badonnel et à moi-même.

— Aux sources devront être ajoutés un README et un Makefile ainsi qu'un rapport décrivant vos choix d'implantation.

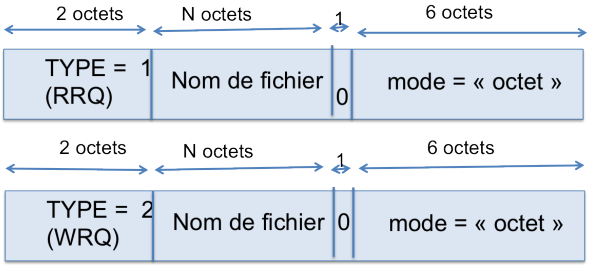
Tout projet non rendu pour le 3 mai 2020 implique que le groupe est considéré comme démissionnaire du module et en conséquence ne pourra pas valider l'UE correspondant pour l'année universitaire 2019-2020.

Toute personne ne se présentant pas à la soutenance sera considérée comme démissionnaire du module et en conséquence ne pourra pas valider l'UE correspondant pour l'année universitaire 2019-2020.

Sujet

Le protocole TFTP est un protocole simple de transfert de fichiers. Il est implanté au dessus de UDP. La fiabilité du transfert des données est assurée au niveau applicatif : un acquittement est envoyé après chaque transfert de données. Une description plus complète du protocole est donnée dans le **RFC1350**. Un message TFTP est de la forme générale :

1. Le client peut effectuer deux types de requêtes : soit écrire un fichier sur le serveur soit récupérer un fichier stocké sur le serveur.



— La valeur du champ `Type` sera égale 1 pour lire un fichier (Read Request : RRQ)

— La valeur du champ `Type` sera égale à 2 pour écrire un fichier (Write Request : WRQ)

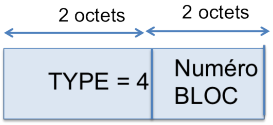
Le champ `Contenu` contient le nom de fichier terminé par un octet égal à 0.

Dans le RFC1350, le mode de transfert est aussi spécifié (netascii or octet). Seul le mode octet sera traité dans le TP (aucune conversion de données entre l'émetteur et le récepteur).

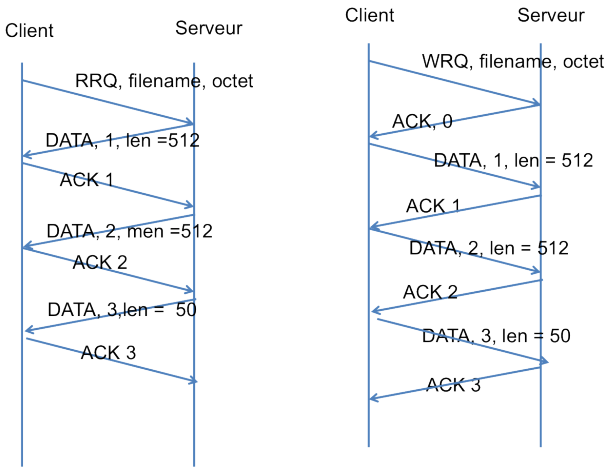
2. Pour le transmission des données, chaque paquet de données doit contenir 512 octets sauf le dernier paquet qui contient entre 0 et 511 octets :

Pour le traitement du dernier paquet, il est recommandé de vous référer au RFC1350 section 6.

3. Le type transmission est *stop-and-wait*. Le numéro de bloc qui est contenu dans chaque message de données est ensuite utilisé dans le message d'acquittement :



Un exemple d'échanges entre le client et le serveur est indiqué ci-après :



Pour une meilleur compréhension du protocole, vous pouvez installer un serveur tftpd sur votre propre machine et observer les messages échangés avec wireshark.

Dans la cadre du TP/Projet, vous devez réaliser un client et un serveur UDP implantant un protocole de transfert de fichiers de type TFTP.

— Le développement se fera par équipe de deux. Un des membres de l'équipe travaillera prioritairement sur le client et l'autre sur le serveur.

— Vous livrerez le code source avec un Makefile permettant la génération des deux exécutables client et serveur et un fichier README donnant les indications pour l'exécution. Le développement doit se faire en langage C sur une machine UNIX. Il peut être réalisé en binôme. Il vous sera demandé de présenter une démonstration de votre travail et de pouvoir expliquer votre code.

— Le client doit pouvoir demander d'écrire ou de lire un fichier sur un serveur. L'adresse IP du serveur, le numéro de port, le nom du fichier ainsi que le type d'opération (lecture ou écriture) devront être fournis par l'utilisateur via la ligne de commande. Pour le projet, il vous est demandé d'implanter prioritairement la fonctionnalité de lecture (RRQ). La fonctionnalité d'écriture (WRQ) est facultative.

— Afin de tester la fiabilité du transfert, vous intégrerez au niveau de votre code un mécanisme de simulation du taux de pertes des données et/ou des acquittements.

— Le code doit pouvoir fonctionner en IPv4 et IPv6 (utilisation de getaddrinfo()).

— Vous n'avez pas à implanter le message ERROR de TFTP. De même, votre serveur tftp n'a pas à gérer plusieurs clients simultanément.