Dans ce document nous expliquons notre avancement dans le projet BE_RESEAU ainsi que le détail de la mise en place de la version 4.1 de MICTCP (fiabilité partielle).

Avancement :

On est arrivés jusqu'à l'implémentation de MICTCP-v4 et plus précisément la version 4.1. Tous les tests ont été passés par les différentes versions (tests texte et vidéo).

Dans le cadre de la version 4.1, nous avons implémenté une amélioration significative du protocole MICTCP. Cette nouvelle version étend la phase de transfert de données MICTCP-v3 en incluant une phase d'établissement de connexion.

De plus, nous avons mis en place un mécanisme de reprise des pertes basé sur le principe du "Stop and Wait". Ce mécanisme permet de garantir une fiabilité partielle en détectant les pertes de données et en les récupérant de manière plus efficace.

Aperçu des fonctionnalités introduites :

Au cours de ce BE, nous avons tout d'abord implémenté une première version de MICTCP qui correspondait à une phase de transfert sans garantie de fiabilité. Cette dernière ressemblait à UDP de part son absence de connexion et de fiabilité. La source envoyait ses données et le puits les recevait de manière disjointe quand il se mettait à l'écoute.

Ensuite, nous avons implémenté la reprise des pertes via le mécanisme "Stop and Wait", lors de la v2. Les deux machines s'échangent alors des acquittements à chaque paquet de données envoyé et renvoie les paquets perdus si l'acquittement n'arrive pas. Il garantit une fiabilité totale et implique également une connexion lors de l'échange, même sans phase d'établissement de cette dernière.

Lors de la v3, nous avons ajouté de la fiabilité partielle à MICTCP afin de répondre au mieux aux contraintes de la transmission vidéo. Le pourcentage de pertes admissibles étant défini de façon statique par le développeur.

Enfin, nous sommes arrivés à la v4.1. Nous avons implémenté une phase d'établissement de connexion, au cours de laquelle les deux machines négocient le pourcentage de pertes admissibles du "Stop and Wait".

Qu’est-ce que la fiabilité partielle :

La fiabilité partielle est une façon d'implémenter le "Stop and Wait" qui va permettre aux machines communicantes de perdre un certain nombre de paquets. Au lieu de bloquer la transmission dès qu'un paquet est perdu, la machine puits va renvoyer un acquittement même si elle n'a pas reçu un paquet et ainsi continuer la transmission tant que le taux de pertes défini auparavant est respecté.

Importance de la fiabilité partielle :

La fiabilité partielle est un facteur indispensable au bon fonctionnement d'un programme devant gérer une transmission vidéo. En effet, beaucoup de paquets de données sont transmis pour une vidéo, et la fluidité de la vidéo passera toujours avant la "fiabilité" de cette dernière pour l'utilisateur.

Si à chaque paquet perdu, on devait renvoyer ce paquet jusqu'à son obtention par l'utilisateur (fiabilité totale), la vidéo serait très lente, et les pertes de paquets engendreraient des freezes et pauses au cours de la vidéo. On perd ici tout le confort attendu par un utilisateur lorsqu'il regarde une vidéo et il vaut mieux alors transmettre une vidéo où certaines frames manquent où ne sont pas parfaitement retransmises, qu'une vidéo qui met très longtemps à charger.

Explications de notre implémentation de la fiabilité partielle :

Nous avons mis en place un système de fiabilité partielle fonctionnant avec une fenêtre déroulante. Avec la mise en place d'une variable globale TOLERANCE, on fixe tous les combiens de paquets on accepte une perte. Par exemple avec TOLERANCE=5 on dit qu'on accepte de perdre un paquet tous les 5 paquets.

Notre première idée était de compter les pertes tous les n paquets avec une fenêtre fixe regardant n paquets utilisant un compteur. Si, lors d'une perte on avait déjà eu une perte dans la fenêtre alors on n'acceptait pas la perte de paquet. On s'est rendu compte que si les pertes étaient localisées en fin et début de fenêtre suivante, on allait perdre deux paquets à la suite. C'est pour celà qu'on a mis en place une fenêtre déroulante.

A l'aide d'un compteur initialisé à 0, on compte chaque paquet bien reçu en ajoutant 1 au compteur et sans dépasser la valeur de tolérance. Lorsqu'on constate une perte, on regarde le compteur. Si le compteur est égal à la Tolérance, on accepte la perte car ça veut dire que les n derniers paquets ont bien étés reçus puis on remet le compteur à 0. Si le compteur est inférieur à la valeur de tolérance, alors on demande la retransmission du paquet car cela veut dire qu'on à déjà eu une perte dans les n derniers paquets.

Grâce à ce système de fiabilité partielle on s'assure que deux paquets consécutifs perdus ne seront pas tolérés comme perte (contrairement à l'utilisation d'une fenêtre fixe). Ainsi, si les n (=TOLERANCE) derniers paquets ont bien étés reçus, et seulement à cette condition, on acceptera la perte d'un paquet pour augmenter la fluidité de la vidéo transmise.