

## **Bureau d'étude « Réseau »**

### **Sujet et Modalités d'organisation et d'évaluation**

3ème année MIC/IR - 2018/19

**Lien d'accès au sujet :** <https://urlz.fr/8SfZ>

Enseignants : EF Bonfoh, G Brau, C. Chassot, R Chakroun, S Toufga  
N. Van Wambeke (Thales AS)



## **Concours de robots!**

### **1. Objectif d'apprentissage et pré requis**

#### **1.1. Acquis de l'apprentissage visés**

- Savoir spécifier un protocole sous forme de « machines à états »
- Savoir implémenter un protocole dans une programmation « de bas niveau », ici en langage C

#### **1.2. Cours et UF constituant les pré requis généraux du BE**

- Cours de C de 3MIC – 3MIC Semestre 5 – UF SE, C, Réseaux, BD
- Cours d'Introduction aux réseaux – 3MIC Semestre 5 – UF SE, C, Réseaux, BD
- Cours Applications Internet et Programmation socket – 3MIC Semestre 6 – UF PROG

#### **1.3. Pré requis spécifiques (!! Maîtrise indispensable !!)**

**Langage C** = langage support d'implémentation du protocole MICTCP

**Applications Internet et Programmation socket :**

- API      socket      →      maîtrise

*absolument*

# obligatoire

## Introduction aux réseaux :

- Partie architecture (modèle en couches, service, protocole) et plus spécifiquement l'architecture TCP/IP
- Phase d'établissement de connexion de TCP qui sera :
  - à spécifier sous forme de machines à états
  - à implémenter puis à étendre en vue d'une négociation dynamique du % de pertes admissibles durant la phase d'établissement de connexion
- Mécanismes de reprise des pertes de type « Stop and Wait », qui sera :
  - à spécifier sous forme de machines à états
  - à implémenter puis à étendre pour y intégrer une gestion de la fiabilité partielle

## 2. Sujet du BE + Modalités d'organisation et d'évaluation

### 2.1 Sujet du BE

L'objectif du BE est de concevoir et de développer en langage C un protocole de niveau Transport appelé **MICTCP**-visant à transporter un flux vidéo distribué en temps réel par une application (fournie par les enseignants) appelée **tsock-vidéo** (cf. Figure 1).

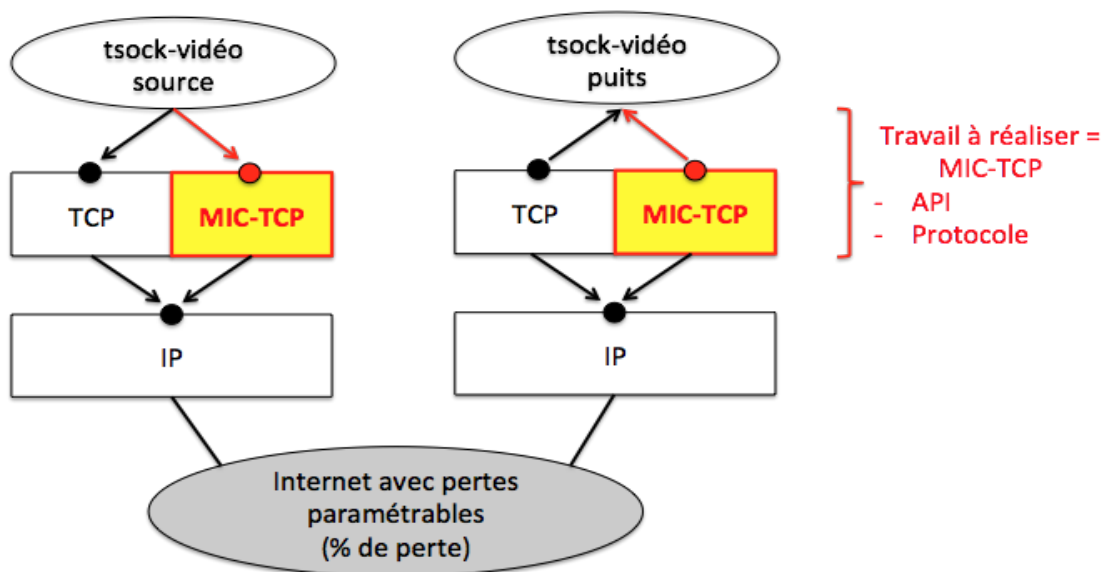


Figure 1 : Positionnement de la contribution attendue

L'application **tsock-vidéo** est basée sur les mêmes spécifications que l'application **tsock** développée dans les TP du cours AIPS du semestre 6 aux différences près suivantes :

- elle génère un flux vidéo et non du texte ;
- elle permet d'invoquer, soit TCP, soit MICTCP comme protocole sous-jacent au transport de la vidéo, ceci par le biais d'une API similaire à l'API socket incluant comme primitives principales (étudiées dans le cours et les TP AIPS) : socket (), bind(), connect(), accept(), send(), recv().

**Ces primitives, dont les signatures seront fournies, devront être développées dans le cadre du BE, faute de quoi l'application tsock-vidéo ne pourra pas s'exécuter.**

**Le protocole MICTCP** devra inclure les phases et mécanismes suivants :

- **une phase d'établissement de connexion** suivant le modèle d'établissement d'une connexion TCP étudié dans le cours d'introduction aux réseaux du semestre 5 ;
- **une phase de transfert de données** unidirectionnelle incluant un mécanisme de reprise des pertes de type « Stop & Wait » tel qu'étudié dans le cours d'Introduction aux réseaux du semestre 5.
- **dans la dernière partie du BE**, le mécanisme de reprise des pertes de type « Stop & Wait » devra être étendu de sorte à inclure un concept nouveau décrit ci-après : celui de gestion d'une fiabilité non plus totale mais partielle. Cette extension conduira à démontrer qu'un transfert de la vidéo via une version de MICTCP « à fiabilité partielle » conduit à une meilleure qualité de la présentation vidéo côté récepteur lorsque le réseau génère des pertes.

**Par soucis de simplification du travail à réaliser dans les temps impartis :**

- MICTCP n'inclura pas de mécanisme de gestion du multiplexage / démultiplexage ;
- MICTCP n'inclura pas de mécanisme de contrôle de flux ;
- MICTCP n'inclura pas de phase de fermeture de connexion ;
- MICTCP assurera un transfert de données en mode message (et non flux d'octets).

**Le concept de gestion d'une fiabilité partielle** est simple à comprendre. Appliqué au BE, il consiste à ce que MICTCP soit autorisé à ne pas délivrer en réception toutes les données applicatives (ici des images) qui lui sont soumises en émission, à condition de respecter un % maximal de pertes admissibles (par exemple : 20% des images peuvent être perdues). Le but est de permettre la délivrance au plus tôt des images contenues dans des PDUs MICTCP reçus hors séquence (c'est à dire pour lesquels le PDU attendu a été perdu suite à une congestion du réseau), tout en respectant le % maximal de pertes admissibles pour la bonne exécution de l'application.

Supposant par exemple que la perte d'une image toutes les 5 images émises soit admissible car non perceptible par l'utilisateur humain.

- 1<sup>er</sup> exemple (Figure 2a) : une image sur 5 est perdue ( $I_4$ ), le PDU MICTCP-véhiculant l'image n'est pas retransmis car la perte est tolérable au regard de la contrainte.

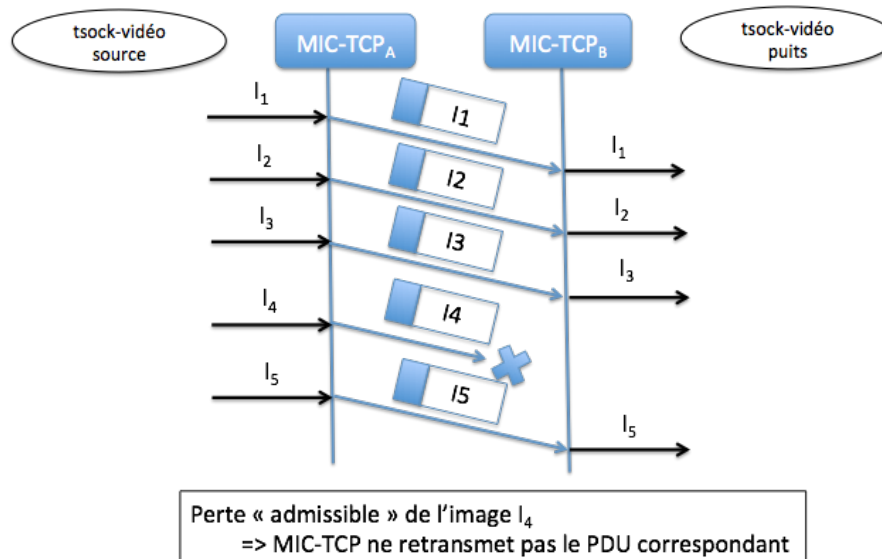


Figure 2a : 1<sup>er</sup> exemple d'application du concept de gestion d'une fiabilité partielle par MICTCP

- 2<sup>ème</sup> exemple (Figure 2b) : les PDUs véhiculant les images  $I_4$  puis  $I_5$  sont perdus. Le 1<sup>er</sup> PDU n'est pas retransmis car la perte est tolérable au regard de la contrainte. Le 2<sup>ème</sup> PDU doit en revanche être retransmis pour satisfaire la contrainte.

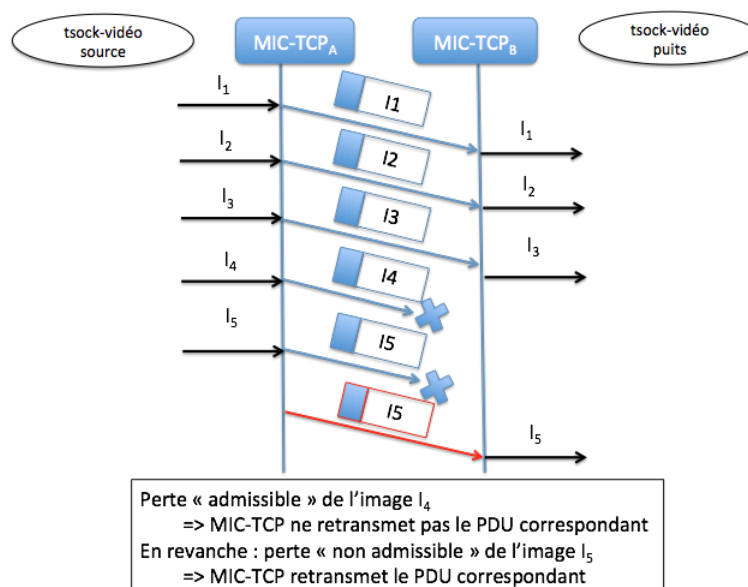


Figure 2b : 2<sup>ème</sup> exemple d'application du concept de gestion d'une fiabilité partielle

## 2.2 Organisation du BE

2 séances de TD suivies de 5 séances de TP :

- Travail individuel en TD
- Travail en binôme en TP

TD 1 orienté compréhension et concepts

- Objectifs :
  - Comprendre le sujet du BE et les modalités du travail à réaliser
  - Comprendre et mettre en œuvre le formalisme des machines à états pour la modélisation des protocoles (ici MICTCP)

TD 2 orienté algorithmique et implémentation

- Objectif :
  - Comprendre les structures et les fonctions mises à disposition pour implémenter le protocole MICTCP
    - structure des PDU MICTCP
    - fonctions de mise à jour des champs de l'entête des PDU
    - fonctions d'envoi et de réception via le service fourni par IP
  - Elaborer les algorithmes nécessaires :
    - à l'établissement d'une connexion
    - à la mise en œuvre d'un mécanisme de reprise des pertes suivant le modèle du « Stop & Wait » (sans gestion de la fiabilité partielle)

TP1 : Implémentation de MICTCP-v1

- Objectifs du TP 1
  - démontrer le besoin d'un protocole plus évolué que TCP pour transporter la vidéo dans de bonnes conditions
  - implanter une version 1 de MICTCP incluant :
    - une phase de transfert de données sans garantie de fiabilité

TP2 et 3 : Implémentation de MICTCP-v2

- Objectif = implanter une version 2 de MICTCP étendant la phase de transfert de données MICTCP-v1 de sorte à inclure :
  - une garantie de fiabilité totale via un mécanisme de reprise des pertes de type « Stop and Wait »

TP4 : Implémentation de MICTCP-v3

- Objectif = implanter une version 3 de MICTCP étendant la phase de transfert de données MICTCP-v2 de sorte à inclure :
  - une garantie de fiabilité partielle « statique » via un mécanisme de reprise des pertes de type « Stop and Wait » à fiabilité partielle « pré

câblée », i.e. dont le % de pertes admissibles est défini de façon statique

TP5 (et au delà en travail personnel si nécessaire) = Implémentation de MICTCP-v4

- Objectifs
  - Implanter une version 4.1 de MICTCP étendant la phase de transfert de données MICTCP-v3 de sorte à inclure :
    - une phase d'établissement de connexion
    - une garantie de fiabilité partielle via un mécanisme de reprise des pertes de type « Stop and Wait » dont le % de pertes admissibles est à présent négocié durant la phase d'établissement de connexion
  - Montrer les bénéfices de MICTCP-v4 comparativement à TCP ou MICTCP-v2
  - Implanter une version 4.2 de MICTCP assurant la gestion de l'asynchronisme entre l'application et MICTCP côté serveur de la connexion et côté récepteur, par le biais du multithreading.

## 2.3 Evaluation du BE

TD non notés mais indispensables au BE => **présence obligatoire des étudiants**

TP notés à l'issue du BE organisés de sorte à permettre :

- aux étudiants : de monter en compétence progressivement
- aux enseignants : **durant le BE et en fonction de l'avancement du travail réalisé en séance**, de réorienter les objectifs à atteindre par les étudiants pour acquérir :
  - le niveau « Maîtrisé » (pour cette partie de l'UF) => version MICTCP-v4
  - le niveau « Acquis » => version MICTCP-v3
  - le niveau « Acquis » en session 2 => version MICTCP-v2
  - le niveau « Partiellement Acquis » ou « Non Acquis » en session 2 => version MICTCP-v1 ou en deçà

## Annexe : Mémo sur l'utilisation de l'archive mictcp fournie :

Dans l'archive MICTCP-2017-18.zip que vous aurez à télécharger lors du 1er TP (disponible sous Moodle ou bien au lien suivant : [goo.gl/GjgzDU](http://goo.gl/GjgzDU)), vous trouverez en particulier le squelette du projet mictcp.c (introduit dans le TD2) qu'il vous est demandé de compléter pour développer les fonctionnalités nécessaires au BE.

Le dossier MICTCP-2017-18 comporte à sa racine 3 fichiers et 3 répertoires :

- fichiers : Makefile, tsock\_texte et tsock\_video
- répertoires : include, src et video

## 1) A la racine du dossier MICTCP-2017-18 se trouve un Makefile, ainsi que deux scripts (.exe) utilisés pour tester votre travail : "tsock\_texte" et "tsock\_video"

La commande "make" permet ainsi de **compiler** le projet dans son ensemble. Il est aussi possible d'utiliser "make prof" pour **générer une archive tar.gz pour les rendus** des différentes versions.

Le script "tsock\_texte" permet d'effectuer des **échanges textuels** utilisant TCP entre un puit et une source.

- Son utilité est de vous permettre de tester votre programme de façon plus rapide qu'avec le script "tsock\_video". Il est donc à utiliser lors de la phase de test et débogage de votre programme.

Le script "tsock\_video" est utilisé pour **transmettre la vidéo** "video/video.bin" entre un puit et une source.

- Ce script sera à utiliser lors du TP1 pour mettre en évidence les limites de TCP dans le transport de la vidéo lorsque le réseau génère des pertes.
- Il sera à utiliser pour la validation de vos versions successives du protocole mictcp.
- L'option -t permet de choisir si vous voulez utiliser le **protocole tcp ou mictcp** pour le transfert.
- Pour arrêter le script, **appuyez sur entrer** (ne pas utiliser ctrl+c).

## 2) Le dossier "include" contient l'ensemble des headers du projet

L'ensemble des **structures à utiliser et des fonctions à implémenter** sont définies dans le fichier "mictcp.h".

- Vous n'avez pas besoin de modifier ce fichier sauf au besoin pour y ajouter des états dans l'énumération "protocol\_state" (cf. TD 1).

Le fichier "api/mictcp\_core.h" contient les headers de certaines **fonctions nécessaires à l'implémentation de mictcp**.

- Vous n'avez pas besoin de modifier ce fichier.

## 3) Le dossier "src"

Le dossier "src" contient la liste des programmes sources du projet.

**Votre travail consiste à modifier le fichier "mictcp.c"** pour développer l'implémentation du protocole mictcp.

Vous pouvez jeter un coup d'oeil dans le dossier "apps" qui contient le code des programmes utilisant votre implémentation de mictcp. NB : ces programmes sont ceux appelés par les scripts "tsock\_texte" et "tsock\_video"

Le fichier "api/mictcp\_core.c" contient le corps des fonctions définis dans "include/api/mictcp\_core.h" et ne doit pas être modifié.

#### **4) Le dossier "video"**

Ce dossier contient quelques vidéo que vous pourrez échanger via le script tsock-video.

Z





Coin <3