UNIVERSITÉ DE ROUEN	ANNÉE 2015-2016
FACULTÉ DES SCIENCES	RÉSEAUX
LICENCE INFORMATIQUE 3E ANNÉE	FICHE DE TP N° 4

Primitives d'adressage pour sockets `AF_INET`

Le but de ce TP est de créer une bibliothèque de manipulation de l'adressage pour des sockets internet. Cette bibliothèque sera utilisée dans tous les prochains TP.

Organisation de votre bibliothèque Pensez à organiser proprement votre bibliothèque : fichier d'entête commenté, séparation des fonctions publiques distribuées par votre bibliothèque et des fonctions d'usage interne, compilation d'une bibliothèque statique ou dynamique, etc.

Exercice 1 Définir un type `AdresseInternet` permettant de manipuler une adresse du domaine internet.

Exercice 2 Écrire les fonctions

1) `AdresseInternet *AdresseInternet_new(const char* adresse, uint16_t port)`

2) `AdresseInternet *AdresseInternet_any(uint16_t port)`

3) `AdresseInternet *AdresseInternet_loopback(uint16_t port)`

qui allouent, remplissent et retournent une adresse internet à partir d'une éventuelle adresse (sous forme DNS ou IP) et d'un numéro de port.

La fonction `AdresseInternet_any` (resp. `AdresseInternet_loopback`) construit une adresse internet correspondant à toutes les interfaces réseau (resp. l'interface loopback) à partir d'un numéro de port.

Valeur de retour : un pointeur sur l'adresse en cas de succès. En cas d'erreur, les fonctions renvoient `NULL`.

Exercice 3 Écrire la fonction

`void AdresseInternet_free(AdresseInternet *adresse)`

qui libère la mémoire qui a été allouée dynamiquement par les fonctions précédentes.

Exercice 4 Écrire un programme qui teste chacune des fonctions de votre bibliothèque en utilisant la bibliothèque créée.

Exercice 5 Écrire la fonction

`int AdresseInternet_getinfo(AdresseInternet *adresse, char *nomDNS, int tailleDNS, char *nomPort, int taillePort)`

qui extrait d'une adresse internet l'adresse réseau et transport correspondants. L'argument `adresse` pointe vers un buffer contenant une adresse internet. L'argument `nomDNS` (resp. `nomPort`) pointe vers un buffer alloué de taille au moins `tailleDNS` (resp. `taillePort`) dans lequel la fonction va écrire une chaîne de caractère (terminant par un 0) contenant le nom (resp. le port) associé à l'adresse fournie. Lorsque cela est possible, la résolution de nom est faite et éventuellement `adresse` est mise à jour.

Si `nomDNS` ou `nomPort` est NULL, l'extraction correspondante ne sera pas effectuée. Les deux ne peuvent pas être NULL en même temps.

Valeur de retour : rend 0 en cas de succès, et -1 en cas d'erreur.

Exercice 6 Écrire la fonction

`int AdresseInternet_getIP(const AdresseInternet *adresse, char *IP, int tailleIP)`

qui extrait d'une adresse internet l'adresse IP correspondante, sous forme de chaîne de caractères.

Valeur de retour : rend 0 en cas de succès, et -1 en cas d'erreur.

Exercice 7 Écrire la fonction

`uint16_t AdresseInternet_getPort(const AdresseInternet *adresse)`

qui extrait le numéro de port d'une adresse internet. L'argument `adresse` se comporte comme dans la fonction précédente.

Valeur de retour : renvoie le port en cas de succès, et 0 en cas d'erreur (par exemple si `adresse` n'est pas initialisée).

Exercice 8 Écrire la fonction

`int AdresseInternet_getDomain(const AdresseInternet *adresse)`

qui rend le domaine de communication de l'adresse internet passée en paramètre (`AF_INET` ou `AF_INET6`).

La fonction rend -1 si l'adresse n'est pas initialisée.

Exercice 9 Écrire les fonctions de convertion

`int sockaddr_to_AdresseInternet(const struct sockaddr *addr, AdresseInternet *adresse)`

et

`int AdresseInternet_to_sockaddr(const AdresseInternet *adresse, struct sockaddr *addr)`

qui construisent une adresse internet à partir d'une structure `sockaddr` (et réciproquement). La structure `addr` doit être suffisamment grande pour pouvoir accueillir l'adresse.

Valeur de retour : 0 en cas de succès, -1 en cas d'échec.

Exercice 10 Écrire la fonction

`int AdresseInternet_compare(const AdresseInternet *adresse1, const AdresseInternet *adresse2)`

qui compare deux adresse internet `adresse1` et `adresse2`.

Valeur de retour : rend 0 si les adresses sont différentes, 1 si elles sont identiques (même IP et même port), et -1 en cas d'erreur.

Exercice 11 Écrire la fonction

`int AdresseInternet_copy(AdresseInternet *adrdst, const AdresseInternet *adrsrc)`

qui copie l'adresse internet `adrsrc` dans la variable pointée par `adrdst` (qui doit être alloué au préalable).

Valeur de retour : rend 0 en cas de succès, une valeur négative en cas d'échec. d'erreur.