

Université de Bretagne Occidentale
Licence Informatique option informatique

Examen de Réseaux

Lundi 8 Septembre 2003, 15h45-17h45

Documents autorisés

Notes de cours, de travaux dirigés et de travaux pratiques.

Remarques

La présentation des réponses est un élément déterminant tant pour la conception de celles-ci que pour leur compréhension par le correcteur. Les réponses doivent être justifiées.

Le barème et le temps sont donnés à titre indicatif.

Lire le sujet en entier avant de commencer l'examen.

1 Question de Cours (6 points - 20 min)

1. Quels sont les intérêts des modèles en couche?
2. Comparer les protocoles TCP et UDP?

2 Taille des buffers (5 points - 20 min.)

2.1 Cas TCP

Soient deux programmes `serveur_tcp` et `client_tcp` communiquant à l'aide du protocole TCP. Le programme `serveur_tcp` envoie de manière régulière des informations au programme `client_tcp`, après la phase initiale de connexion.

2.1.1 extrait de code de `serveur_tcp`

```
#define BUFSIZE_SERVEUR 21
...

char buf[BUFSIZE_SERVEUR];
...

strcpy(buf, "01234567899876543210");
if (write(sock, buf, strlen(buf)+1) != strlen(buf)+1)
    { ... }
...

```

2.1.2 extrait de code de `client_tcp`

```
#define BUFSIZE_CLIENT 21
...

char buf[BUFSIZE_CLIENT];
...

if (read(socket_service, buf, BUFSIZE_CLIENT) < 0)
    { ... }
printf("j'ai reçu %s\n", buf);
...

```

Que se passe-t-il dans le cas où la variable `BUFSIZE_CLIENT` a une valeur plus grande que la variable `BUFSIZE_SERVEUR`? Peut-on gérer facilement les éventuels problèmes? Si oui comment?

Que se passe-t-il dans le cas où la variable `BUFSIZE_CLIENT` a une valeur plus petite que la variable `BUFSIZE_SERVEUR`? Peut-on gérer facilement les éventuels problèmes? Si oui comment?

2.2 Cas UDP

Soient deux programmes `emetteur_udp` et `receveur_udp` communiquant à l'aide du protocole UDP. Le programme `emetteur_udp` envoie de manière régulière des informations au programme `receveur_udp`.

2.2.1 extrait de code de `emetteur_udp`

```
#define BUFSIZE_EMETTEUR 21
...

char test[BUFSIZE_EMETTEUR];
...

strcpy(test, "01234567899876543210");
if ((envoye = sendto(sock, test, strlen(test)+1, 0, &adr, lgadr)) != strlen(test)+1)
    { ... }
...
```

2.2.2 extrait de code de `receveur_udp`

```
#define BUFSIZE_RECEVEUR 21
...

char test[BUFSIZE_RECEVEUR];
...

if ((recu = recvfrom(sock, test, BUFSIZE_RECEVEUR, 0, &adr, &lgadr)) == -1)
    { ... }
printf("j'ai recu %s\n", test);
...
```

Que se passe-t-il dans le cas où la variable `BUFSIZE_RECEVEUR` a une valeur plus grande que la variable `BUFSIZE_EMETTEUR`? Peut-on gérer facilement les éventuels problèmes? Si oui comment?

Que se passe-t-il dans le cas où la variable `BUFSIZE_RECEVEUR` a une valeur plus petite que la variable `BUFSIZE_EMETTEUR`? Peut-on gérer facilement les éventuels problèmes? Si oui comment?

3 Problème (9pts - 50 min.)

On se propose de construire un système de **Chat** multi-utilisateurs.

- Présentez et justifiez des choix architecturaux pour un tel système.
- Proposez une implémentation à haut niveau (langage de description algorithmique).
- Décrivez en détail votre **Chat**.