Examen Client/Serveur, 2002/2003

DESS CCI option Réseau

Durée : 1 heure

Tout document autorisé

Faites UN SEUL choix par question. +1 points si réponse juste, -0.5 si réponse fausse, 0 sinon.

Si plusieurs réponses sont possibles, prenez la plus précise. Par exemple :

- *23 est un nombre* : **(A)** pair, **(B)** impair, **(C)** premier.

La réponse que j'attends est *premier*, la réponse *impair* sera considérée comme fausse.

1. *Un socket en mode connecté permet* : **(A)** D'envoyer un message, **(B)** De recevoir un message, **(C)** De dialoguer.
2. *Un fois que le client est connecté avec un socket au serveur* : **(A)** Il envoie un message au serveur, **(B)** Il attend un message du serveur, **(C)** Cela dépend.
3. *Quand un client plante, le serveur* : **(A)** Plante, **(B)** Ferme le socket du client, **(C)** Attend que le socket redevienne actif.
4. *Avec un socket en mode connecté, quand un routeur s'arrête 2 secondes* : **(A)** La liaison socket est rompue, **(B)** Le client ferme le socket, **(C)** Personne ne s'en aperçoit.
5. *Un serveur basé sur les sockets plante, les clients* : **(A)** Ne s'en aperçoivent pas, **(B)** Sont automatiquement arrêtés, **(C)** S'en aperçoivent quand ils veulent dialoguer.
6. *Les RPC sont programmés en utilisant* : **(A)** Des sockets, **(B)** Des packets IP, **(C)** Directement la carte réseau.
7. *Les RPC permettent* : **(A)** De dialoguer en binaire non lisible, **(B)** De faire de l'asynchrone, **(C)** De transférer des données énormes.
8. *Un serveur RPC ne permet pas* : **(A)** D'avoir plusieurs fonctions différentes, **(B)** De diffuser des versions différentes d'un même protocole, **(C)** De faire un protocole avec état.
9. *10000 clients transfèrent des gros fichiers sur un unique serveur FTP, le goulot d'étranglement est* : **(A)** Le débit réseau du serveur, **(B)** Le débit disque du serveur, **(C)** Le processeur du serveur.
10. *Quand un serveur NFS plante et redémarre, les clients* : **(A)** Ne s'en aperçoivent pas, **(B)** Doivent relancer le service NFS, **(C)** Doivent tuer les applications bloquées.
11. *Pour lancer un serveur Web, il faut* : **(A)** Être `root`, **(B)** Utiliser un port supérieur à 1024, **(C)** Avoir un serveur supportant le protocole HTTP.
12. *Quand on se connecte à un port inférieur à 1024, cela garantit que le serveur* : **(A)** Est sécurisé, **(B)** Est lancé par `root`, **(C)** Est digne de confiance.
13. *On fait du client/serveur pour* : **(A)** Accélérer les traitements, **(B)** Partager les ressources, **(C)** Faire des programmes portables.
14. *On ne fait pas de client/serveur* : **(A)** Si les machines sont très éloignées, **(B)** Si le réseaux est à faible débit, **(C)** Si l'application est inutilisable.
15. *On veut faire du calcul parallèle* : **(A)** Il y a beaucoup de serveurs et 1 client, **(B)** Il y a beaucoup de clients et 1 serveur, **(C)** Il n'y a qu'un seul serveur par machine.
16. *On utilise le proxy/cache pour* : **(A)** Diminuer l'espace disque sur le serveur, **(B)** Accélérer les chargements des tout petits fichiers, **(C)** Diminuer la bande passante longue distance utilisée.
17. *On lance les serveurs avec* `inetd` *quand on veut* : **(A)** Un accès rapide au serveur, **(B)** Un accès fiable au serveur, **(C)** Moins de sockets ouverts simultanément.
18. *On ne peut pas faire de client/serveur* : **(A)** Si on n'a pas TCPIP, **(B)** Si on n'a pas de Socket, **(C)** Si on n'a pas moyen de communication.
19. *Vous tapez des commandes UNIX dans une fenêtre X11, il y a combien de clients et serveurs qui interviennent* : **(A)** ≤ 1, **(B)** 2, **(C)** ≥ 3.
20. *Le noyau du système d'exploitation* : **(A)** N'est client d'aucun serveur, **(B)** Est serveur de ressources pour tous les processus, **(C)** Ne peut pas être son propre client.

Correction

1. (C) : sinon il n'y aurai pas de question réponse
2. (C) : (A) est faux pour FTP, (B) est faux pour HTTP
3. (B) : (A) s'il plante il y a un bug, (C) on ne peut pas reconnecter un socket déconnecté.
4. (C) : (A) le socket est rompu quand on recontacte la machine et qu'elle nous dit qu'il n'existe pas, (B) le client femre le socket quand il est certain que cela ne marchera pas (plusieurs minutes souvent).
5. (C) : (A) ils s'en apper coivent forcément sinon cela veut dire qu'ils n'ont pas besoin du serveur, (B) Par qui ?
6. (A) : (B) cela serait stupide de tout reprogrammer, (C) la bibliothèque RPC devrait connaîtoutes les cartes réseau.
7. (A) : (B) non, on est obligé de faire des *threads*, (C) impossible de transféré des données ne tenant pas dans la mémoire.
8. (pas de bonne réponse) : (A) c'est possible (voir TP), (B) c'est possible voir cours, (C) c'est possible (variables globales ou statiques).
9. (A) : le débit disque est de plusieurs dizaines de Mo/s par disque et il peut y avoir plusieurs interfaces, le débit réseau couramment de 10Mo/s par interface. Le CPU ne fait pas grand chose, ce ne sont que des transferts, cela ne serai pas le cas si c'était de nombreux petits fichiers.
10. (A) : (B) pas la peine, les requêtes RPC ne vont retourner leur réponse que quand le serveur aura redémarré, (C) Les applications se débloquent quand le serveur redémarre.
11. (C) : (A) vous pouvez utiliser un port >1024, (B) pas si vous êtes *root*.
12. (B) : (A) non, il peut y avoir plein de faille de sécurité, (C) non, ce n'est pas forcément qui vous croyez.
13. (B) : (A) si le réseau est lent cela ralenti, (C) cela ne rend pas les programmes portable, au contraire ils utilises des ressources systèmes supplémentaires.
14. (C) : (A) google.com (B) 1bit/s suffit pour commander sa domotique.
15. (A) : (B) ca c'est du partage de temps CPU, c'est plus lent, (C) on a tout intérêt à mettre autant de serveur qu'il y a de processeurs dans la machine.
16. (C) : (A) c'est exactement le même espace, (B) pour les petits fichier l'augmentation de latence fait perdre les avantages.
17. (B) : (A) le premier accès est plus lent, (C) *inetd* ouvre tous les sockets donc le nombre est le même.
18. (C) : (A) on fait du client/serveur en dialoguant avec une imprimante non-réseau, (B) itou
19. (C) : il y a le **shell** qui interprète les commandes, il demande au **kernel** de lire le clavier, mais celui-ci est géré par **xterm** qui demande à lire le clavier à **X11** qui lui même récupè l'information via le **window manager**. Cela fait donc 5 intervenants.
20. (B) : (A) Il est client RPC pour le service NFS, (C) Il est indirectement son propre client car il peut demander des services à des processus normaux.