Institut Galilée — Systèmes d'exploitation, Réseaux
Année 2013-2014 — Licence 2

Contrôle Systèmes d'exploitation, Réseaux

Vendredi 4 Juillet 2014 — 9h - 12h
Aucun document n'est autorisé

Exercice 1 (4 = 2 + 2) *On considère 5 programmes P_1, P_2, P_3, P_4, P_5 de début d'exécution respectif aux temps $t_0, t_0+1, t_0+3, t_0+5, t_0+7$ et de durée d'exécution respective $4, 4, 2, 4, 2$. On suppose que l'ordinateur a un seul CPU. Donner les diagrammes de Gantt et les temps moyens de traitement pour des ordonnancements de type FIFO (premier arrivé premier servi) et à tourniquet (pour un quantum de 2).*

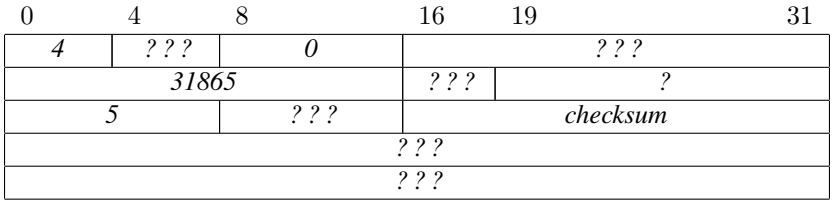
Exercice 2 (4 = 2 + 2)

1. *Ecrivez un programme dans lequel le processus père crée 3 processus fils. Chaque processus fils attend 3 secondes, puis envoie au processus père le signal SIGUSR2, puis termine. Le père termine lorsqu'il a reçu les trois signaux. (on n'écrira pas les* `include` *nécessaires)*
2. *En utilisant le mécanisme des tubes, modifiez le programme de telle sorte que chaque fils, après 3 secondes, envoie son pid par un tube au père, et que celui-ci affiche alors le pid reçu.*

Exercice 3 (5 = 1 + 1 + 1 + 1 + 1) *Une entreprise composée de 9 départements se voit affecter l'adresse IP 202.88.181.0. L'administrateur souhaite affecter un sous-réseau à chaque département.*

1. *De quelle classe d'adressage s'agit-il ? Combien de machines cela permet-il d'adresser ?*
2. *En supposant que le nombre de départements de l'entreprise ne va pas tellement évoluer, quel est le masque de sous-réseau optimal ?*
3. *Combien de départements peuvent être ajoutés avec ce masque ?*
4. *Combien de machines chaque département peut-il comporter ?*
5. *Quelle est l'adresse de broadcast du premier sous-réseau*

Exercice 4 (3) *Une station X souhaite transmettre à une autre station Y un datagramme UDP dont la taille (incluant l'entête UDP) est égale à 2000 octets. La machine X est dans un réseau A alors que Y est dans un réseau B. L'adresse de X est 154.13.54.6, celle de Y est 223.33.17.45. La MTU du réseau A est égale à 1024 octets. Le datagramme envoyé par X quitte le réseau A en passant par un routeur R1, il atteint le réseau B de MTU = 2048 octets. La structure de l'en-tête IP du datagramme dans le réseau A est présentée ci-dessous :*



Déterminez les paquets IP (en précisant l'en-tête de chaque paquet) qui passent dans les réseaux A et B. (La somme de contrôle de l'en-tête n'est pas à calculer)

Exercice 5 (4 = 1 + 1 + 2)

Il s'agit d'implémenter un chat simplifié en utilisant le protocole TCP. Le serveur est censé accepter deux clients et les "mettre en contact" : tout ce qui est reçu sur la socket de service de l'un est réémis sur la socket de service de l'autre. Le serveur est à l'écoute sur le port 3333 et a le numéro IP 202.33.177.154

1. *Donner la succession des fonctions système permettant au serveur d'être à l'écoute, commenter chacune de ces opérations.*
2. *Ecrire le code du serveur.*
3. *Ecrire le code des clients.*

(Annexe 1) Fonctions et structures utiles :

```
unsigned int sleep(unsigned int seconds);
pid_t fork(void);
int pipe(int tableau[2]);
pid_t getpid(void);
void (* signal(int signum, void (*handler) (int))) (int);

int socket(int domaine, int type, int protocole);
int bind(int sock, struct sockaddr * adr, socklen t size);
int listen(int sock, int backlog) ;
int accept(int sock, struct sockaddr *adresse, socklen_t *longueur) ;
int connect(int sock, struct sockaddr *addr serv, socklen t longueur) ;
int close(int fd);
ssize_t read(int fd, void *buf, size_t count);q
ssize_t write(int fd, const void *buf, size_t count);
uint32_t htonl(uint32_t hostlong);
uint16_t htons(uint16_t hostshort);
uint32_t ntohl(uint32_t netlong);
uint16_t ntohs(uint16_t netshort);
void bzero(void *s, size_t n);
int inet aton(const char *cp, struct in_addr *in) ;
char * inet ntoa(struct in_addr in) ;
struct hostent * gethostbyname(char * hostname) ;
struct sockaddr {
   short sa family ;
   char sa data[14] ;
} ;
struct sockaddr in {
   short sin_family ;
   u_short sin_port ;
   struct in_addr sin_addr ;
   char sin_zero[8] ;
} ;
struct hostent {
   char * h_name ;
   char **h_aliases ;
   int h_addrtype ;
   int h_length ;
   char ** h_addr list ;
} ;
```

(Annexe 2) Structure d'un paquet IPv4 :