Examen de Protocoles Réseaux

Juliusz Chroboczek

23 juin 2017

La durée de l'examen est de 2 heures. Les documents sont autorisés, le matériel électronique est interdit. Le sujet consiste de 2 pages.

Exercice 1 (Codes). On désire transmettre, sur un canal, un message de n bits de long. Pour des raisons techniques, le canal ne peut transmettre que des blocs de 16 bits, séquentiellement (les blocs arrivent dans l'ordre d'émission). Si cela vous arrange (pour ne pas avoir de *padding*), vous pouvez supposer que n a la taille que vous voulez (mais $n \geq 1024$). Proposez un code efficace (en terme de bande passante) pour chacun des cas suivants (si c'est un code vu en TD il suffit de le nommer). À chaque fois précisez l'impact sur la bande passante.

1. On suppose qu'au plus *un bit* peut être erroné (en tout : un bit dans un seul bloc, pas dans chaque bloc). On veut *détecter* s'il y a eu erreur.
2. On suppose encore qu'au plus *un bit* en tout peut être erroné. Si cela arrive, on veut *corriger* (sans réémission) l'erreur.
3. On suppose que des erreurs de transmission peuvent arriver sur au plus *un des blocs* transmis (au pire, tout un bloc peut être corrompu, mais les autres blocs ne le seront pas). On veut *détecter* si cela est arrivé (inutile de savoir dans quel bloc).
4. On suppose aussi que des erreurs peuvent arriver sur au plus *un des blocs* transmis, et cette fois, on veut *corriger* (sans réémission) les erreurs éventuelles.

Exercice 2 (CRC). On a reçu le message $M = 1010011000001101$ codé par le CRC-8 correspondant au polynôme $X^8 + X^7 + X^3 + X + 1$. Le message reçu est-il correct ? Si oui, donnez le message décodé. Si non, justifiez en une phrase. Dans les deux cas, donnez les étapes du calcul.

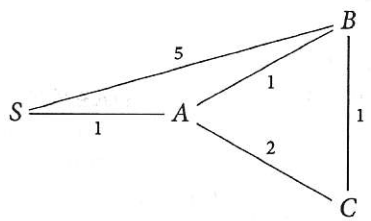
Exercice 3 (Traces). On considère le fragment de trace suivant :

```
00:56:25.827317 IP6 2001:660:3301:9200::51c2:1b9b.40394 > 2a01:4f8:202:43ea::3.443: Flags [.],
    ack 789027, win 2102, options [nop,nop,TS val 627271522 ecr 856931959], length 0
00:56:25.827356 IP6 2a01:4f8:202:43ea::3.443 > 2001:660:3301:9200::51c2:1b9b.40394: Flags [.],
    seq 789027:790455, ack 905, win 33, options [nop,nop,TS val 856931959 ecr 627271516], length 1428
00:56:25.851294 IP6 2a01:4f8:202:43ea::3.443 > 2001:660:3301:9200::51c2:1b9b.40394: Flags [.],
    seq 790455:791883, ack 905, win 33, options [nop,nop,TS val 856931965 ecr 627271522], length 1428
00:56:25.851298 IP6 2001:660:3301:9200::51c2:1b9b.40394 > 2a01:4f8:202:43ea::3.443: Flags [.],
    ack 791883, win 2209, options [nop,nop,TS val 627271528 ecr 856931959], length 0
00:56:25.851304 IP6 2a01:4f8:202:43ea::3.443 > 2001:660:3301:9200::51c2:1b9b.40394: Flags [.],
    seq 791883:793311, ack 905, win 33, options [nop,nop,TS val 856931965 ecr 627271522], length 1428
00:56:25.851319 IP6 2a01:4f8:202:43ea::3.443 > 2001:660:3301:9200::51c2:1b9b.40394: Flags [.],
```

```
    seq 793311:794739, ack 905, win 33, options [nop,nop,TS val 856931965 ecr 627271522], length 1428
00:56:25.851322 IP6 2001:660:3301:9200::51c2:1b9b.40394 > 2a01:4f8:202:43ea::3.443: Flags [.],
    ack 794739, win 2209, options [nop,nop,TS val 627271528 ecr 856931965], length 0
00:56:25.851329 IP6 2a01:4f8:202:43ea::3.443 > 2001:660:3301:9200::51c2:1b9b.40394: Flags [.],
    seq 794739:796167, ack 905, win 33, options [nop,nop,TS val 856931965 ecr 627271522], length 1428
00:56:25.851367 IP6 2a01:4f8:202:43ea::3.443 > 2001:660:3301:9200::51c2:1b9b.40394: Flags [.],
    seq 796167:797595, ack 905, win 33, options [nop,nop,TS val 856931965 ecr 627271522], length 1428
00:56:25.851371 IP6 2001:660:3301:9200::51c2:1b9b.40394 > 2a01:4f8:202:43ea::3.443: Flags [.],
    ack 797595, win 2209, options [nop,nop,TS val 627271528 ecr 856931965], length 0
00:56:25.851379 IP6 2a01:4f8:202:43ea::3.443 > 2001:660:3301:9200::51c2:1b9b.40394: Flags [P.],
    seq 797595:799023, ack 905, win 33, options [nop,nop,TS val 856931965 ecr 627271522], length 1428
00:56:25.852907 IP6 2a01:4f8:202:43ea::3.443 > 2001:660:3301:9200::51c2:1b9b.40394: Flags [.],
    seq 799023:800451, ack 905, win 33, options [nop,nop,TS val 856931966 ecr 627271522], length 1428
00:56:25.852913 IP6 2001:660:3301:9200::51c2:1b9b.40394 > 2a01:4f8:202:43ea::3.443: Flags [.],
    ack 800451, win 2209, options [nop,nop,TS val 627271529 ecr 856931965], length 0
00:56:25.852917 IP6 2a01:4f8:202:43ea::3.443 > 2001:660:3301:9200::51c2:1b9b.40394: Flags [.],
    seq 800451:801879, ack 905, win 33, options [nop,nop,TS val 856931966 ecr 627271522], length 1428
00:56:25.852927 IP6 2a01:4f8:202:43ea::3.443 > 2001:660:3301:9200::51c2:1b9b.40394: Flags [.],
    seq 801879:803307, ack 905, win 33, options [nop,nop,TS val 856931966 ecr 627271522], length 1428
00:56:25.852929 IP6 2001:660:3301:9200::51c2:1b9b.40394 > 2a01:4f8:202:43ea::3.443: Flags [.],
    ack 803307, win 2209, options [nop,nop,TS val 627271529 ecr 856931966], length 0
```

De quel protocole de couche application s'agit-il probablement ? Dans quelle direction se fait le transfert de données de couche application ? Sur laquelle des deux machines cette trace a-t-elle été capturée ? Quelle quantité de données de couche application a été transférée au total et quel est le débit moyen du transfert à la couche application ?

Exercice 4 (Routage). On considère la topologie suivante, où les entiers sont les coûts des liens :



On s'intéresse au routage à vecteur de distances où la source est fixée à S.

1. Faites évoluer le protocole à vecteur de distances naïf depuis l'état initial jusqu'à convergence. Vous n'avez pas à justifier votre réponse — il suffit de me fournir un tableau ayant la forme suivante :

S	d =?, nh =?	d =?, nh =?	···
A	d =?, nh =?	d =?, nh =?	···
B	d =?, nh =?	d =?, nh =?	···
C	d =?, nh =?	d =?, nh =?	···

2. On se place dans la configuration où l'algorithme a convergé précédemment, et on suppose que le lien de coût 1 entre S et A a cassé (son coût est maintenant infini). Faites évoluer l'algorithme dans la nouvelle topologie jusqu'à convergence. Précisez s'il y a une boucle de routage, et, si c'est le cas, les étapes pendant lesquelles elle a lieu.