**UNIVERSITÉ DE MONTRÉAL**

DEPARTEMENT D’INFORMATIQUE ET DE RECHERCHE OPÉRATIONNELLE

**IFT-3325 TELE INFORMATIQUE**

**Devoir 3 - Exercices**

Louis BERTRAND, Matricule : 20141254

Ronald GEDEON, Matricule : 20142399

Fig.1 : Identification des étudiants.

Professeur Abdelhakim HAFID

**Exercice 1** (10 points)

1. (6 points) **Un serveur web est généralement mis en place pour recevoir relativement des petits messages de ses clients et transmettre potentiellement des messages de grandes tailles. Expliquer quel type de protocole ARQ (rejet sélectif, Go-Back-N) entraînera moins de charge à ce serveur.**

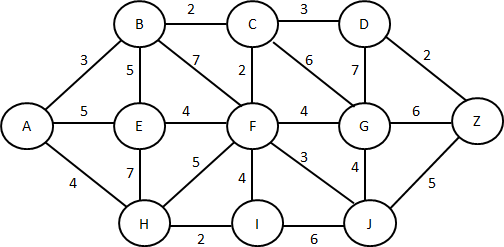
Go-Back-N, car le serveur n’aura pas besoin de grand tampon/buffer pour mémoriser. Mais, si les erreurs sont fréquentes, Selective-Reject serait plus efficace pour optimiser le bandwith dans l’utilisation du canal.

1. (4 points) **L’utilisation de la même valeur du fanion au début et à la fin de la trame peut causer des problèmes si un bit est permuté. Décrire quelques exemples de problèmes qui peuvent se produire.**

On peut avoir des erreurs de délimitation de trames, donc aussi des erreurs de transmission de trames.

**Exercice 2** (16 points)

**Appliquer l’algorithme de routage Dijkstra et Bellman Ford pour trouver le plus court chemin du nœud A jusqu’au nœud Z de la topologie suivante (inclure toutes les étapes du calcul).**



Algorithme de Dijkstra si on part de A

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| T | L(B) | Path | L(C) | Path | L(D) | Path | L(E) | Path | L(F) | Path | L(G) | Path | L(H) | Path | L(I) | Path | L(J) | Path | L(Z) | Path |
| A | 3 | A | ∞ | - | ∞ | - | 5 | A | ∞ | - | ∞ | - | 4 | A | ∞ | - | ∞ | - | ∞ | - |
| AB | 3 | A | 5 | B | ∞ | - | 5 | A | ∞ | - | ∞ | - | 4 | A | ∞ | - | ∞ | - | ∞ | - |
| AB | 3 | A | 5 | B | ∞ | - | 5 | A | 9 | H | ∞ | - | 4 | A | 6 | H | ∞ | - | ∞ | - |
| ABC | 3 | A | 5 | B | 8 | C | 5 | A | 7 | C | 11 | C | 4 | A | 6 | H | ∞ | - | ∞ | - |
| ABC | 3 | A | 5 | B | 8 | C | 5 | A | 7 | C | 11 | C | 4 | A | 6 | H | ∞ | - | ∞ | - |
| ABC | 3 | A | 5 | B | 8 | C | 5 | A | 7 | C | 11 | C | 4 | A | 6 | H | 12 | I | ∞ | - |
| ABCD | 3 | A | 5 | B | 8 | C | 5 | A | 7 | C | 11 | C | 4 | A | 6 | H | 10 | F | ∞ | - |
| ABCDZ | 3 | A | 5 | B | 8 | C | 5 | A | 7 | C | 11 | C | 4 | A | 6 | H | 10 | F | 10 | D |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |

Il n’est pas nécessaire de continuer le tableau. On peut conclure à partir de ce que l’on a que le chemin le plus court de A à Z est A,B,C,D,Z.

Algorithme de Bellman Ford

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| h | L(a) | Path | L(b) | Path | L(c) | Path | L(d) | Path | L(e) | Path | L(f) | Path | L(g) | Path | L(h) | Path | L(I) | Path | L(J) | Path | L(z) | Path |
| 0 | 0 | - | ∞ | - | ∞ | - | ∞ | - | ∞ | - | ∞ | - | ∞ | - | ∞ | - | ∞ | - | ∞ | - | ∞ | - |
| 1 | 0 | - | 3 | A | ∞ | - | ∞ | - | 5 | A | ∞ | - | ∞ | - | 4 | A | ∞ | -- | ∞ | - | ∞ | - |
| 2 | 0 | - | 3 | A | 5 | AB | ∞ | - | 5 | A | 9 | AE | ∞ | - | 4 | A | 6 | AH | ∞ | - | ∞ | - |
| 3 | 0 | - | 3 | A | 5 | AB | 8 | ABC | 5 | A | 7 | ABC | 11 | ABC | 4 | A | 6 | AH | 12 | AHI | ∞ | - |
| 4 | 0 | - | 3 | A | 5 | AB | 8 | ABC | 5 | A | 7 | ABC | 11 | ABC | 4 | A | 6 | AH | 10 | ABCF | 10 | ABCD |
| 5 | 0 | - | 3 | A | 5 | AB | 8 | ABC | 5 | A | 7 | ABC | 11 | ABC | 4 | A | 6 | AH | 10 | ABCF | 10 | ABCD |
| 6 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| 7 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| 8 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| 9 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |

Après la cinquième itération, on remarque qu’il n’y a aucun changement dans le tableau. On peut donc s’arrêter là et conclure que le chemin optimal entre A et Z est A,B,C,D,Z.

**Exercice 3** (12 points)

1. (6 points) **Deux réseaux (A et B) utilisent le protocole TCP/IP; ils sont reliés via un routeur. La compagnie a défini le masque sous réseau : 255.255.0.0. Une station du réseau A ayant une adresse 101.64.0.102 se plaint de ne pas**

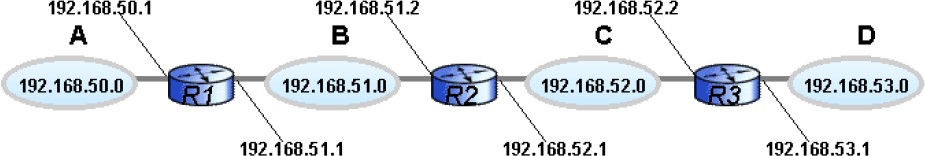
**pouvoir joindre la station, du réseau B, dont l’adresse est 101.64.45.102. Expliquer pourquoi.**

La station du réseau A et la station du réseau B appartiennent au même sous-réseau, puisque l’opération AND entre le masque et leur adresse IP donne l’adresse du sous-réseau. Soit :

* A : 255.255.0.0 AND 101.64.0.102 = 101.64.0.0.
* B : 255.255.0.0 AND 101.64.45.102 = 101.64.0.0.

Par conséquent, pour communiquer les machines doivent utiliser leurs adresses physiques (MAC) et non pas leurs adresses IP.

1. (6 points) **Soit la topologie suivante** :



**et la table de routage de R2**

Gateway

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Adresse | Masque | Passerelle | Interface | Metrique |
| 192.168.50.0 | 255.255.255.0 | 192.168.51.1 | 192.168.51.2 | 2 |
| 192.168.51.0 | 255.255.255.0 | DIRECT | 192.168.51.2 | 1 |
| 192.168.51.2 | 255.255.255.255 | 127.0.0.1 | 127.0.0.1 | 1 |
| 192.168.52.0 | 255.255.255.0 | DIRECT | 192.168.52.1 | 1 |
| 192.168.52.1 | 255.255.255.255 | 127.0.0.1 | 192.0.0.1 | 1 |
| 192.168.53.0 | 255.255.255.0 | 192.168.52.2 | 192.168.51.2 | 2 |
| 255.255.255.255 | 255.255.255.255 | 185.75.142.6 | 185.75.142.6 | 1 |

**A-t-on besoin de modifier quelques informations dans cette table pour que les réseaux A et B communiquent avec le réseau D? Justifier votre réponse.**

Oui, il faudrait modifier la table de routage. En fait, pour aller au sous-réseau A (192.168.50.0), il faut traverser B et A, donc une métrique de 2. Le masque du sous-réseau reste pareil. La passerelle ou Gateway (destination) de R1 reste inchangé, ainsi que l’interface (source) de R2 reste la même.

Le sous-réseau B (192.168.51.0) est le Next-Hop de R2, donc pas besoin de passerelle et la métrique est 1, ainsi que la communication est directe. Le masque du sous-réseau et l’interface de R2 restent pareil.

De même, le sous-réseau C (192.168.52.0) est le Next-Hop de R2, donc pas besoin de passerelle et la métrique est 1, ainsi que la communication est directe. Le masque du sous-réseau et l’interface de R2 n’ont pas besoin de changer.

Pour aller au sous-réseau D (192.168.53.0), il faut traverser C et D, donc une métrique de 2. Le masque du sous-réseau reste pareil. La passerelle ou Gateway (destination) de R3 reste doit changer pour 192.168.52.2, ainsi que l’interface (source) de R2 reste la même.

**Exercice 4** (10 points)

1. (3 points) **Donner et expliquer 3 exemples de paramètres négociables lors de l’établissement d’une connexion au niveau de la couche de transport.**

- Taille de la fenêtre : le nombre de segments maximal qui peut être envoyé d’une entité (application) à un autre sans recevoir de confirmation.

- Taille maximale de segment : la capacite en octets de chaque segment.

- 2 connexions successives : le 2eme doit commencer avec un numéro très éloigné du premier.

1. (4 points) **Donner et expliquer la liste des champs d’un paquet IP susceptibles de se changer lors d’une fragmentation.**

-Total Length: Pour chaque fragment, ça va contenir la taille du fragment (On aura des fragments de longueur différentes alors ce champ variera d’un fragment à l'autre).

- Le champ bit More sera mis à 1 dans tous les fragments et le dernier fragment aura le champ bit More = 0.

- Le champ offset change : position du premier octet utile du fragment par rapport au Premier octet utile du paquet.

1. (3 points) **Donner les avantages et les inconvénients de chacune des situations suivantes :**
   * **Le Réassemblage des paquets s’effectue par les routeurs**

-Avantages : Les pertes de paquets sont détectées plus tôt et peut réduire la fragmentation pour certains groupes de routeurs de grande capacite.

-Inconvénients : n’est pas performant, car augmente le délai considérablement (les paquets suivent des chemins différents et les routeurs sont de taille différente), donc les paquets sont fragmentes plusieurs fois avant d’arriver à destination.

* + **Le Réassemblage des paquets s’effectue à la destination**

-Avantages : réduction de la charge de travail des routeurs et amélioration de la performance du réseau.

Inconvénients : On doit gérer certains problèmes liés à la fragmentation tels que le dé-séquencement des paquets, contrôle de flux et d’erreurs un peu à la dernière minute dans le réseau. Et, si on se rend compte à la destination qu’on a perdu un fragment, on doit redemander le paquet à partir de l’origine.

**Exercice 5** (12 points)

**On suppose qu’on utilise un « two-way » handshake au lieu d’un « three-way » handshake pour l’établissement d’une connexion. Dans ce cas le troisième message n’est pas requis. Est-ce qu’il peut y avoir un blocage (deadlock)? Donner dans ce cas un exemple ou montrez qu’il ne peut y avoir de blocage.**

Oui. Suite a la demande de connexion de A (SYN i), B confirme avec SYN j=0, AN =i (ne m’envoie rien). Par la suite, B envoie une autre confirmation avec une taille de fenêtre w = j, j > 0 et que cependant ce segment est perdu. L’émetteur croit alors qu’il ne peut rien envoyer (w=0) et le récepteur croit le contraire. Solution : temporisateur de segment.

**Exercice 6** (12 points)

**Une entité TCP ouvre une connexion et utilise un « slow start ». Combien de durées aller-retour (round-trip) doivent être écoulées avant que l’entité TCP ne puisse envoyer M octets ; on suppose que la taille d’un segment est égale à N octets.**

Propriété slow-start :

1er envoie = N octets pour 1er aller-retour

2eme envoie = 2∧1 \* N octets pour 2e aller-retour

3eme envoie = 2∧2 \* N octets pour 3e aller-retour

4eme envoie = 2∧3 \* N octets pour 4e aller-retour

…

j\_ieme envoie = 2^(j-1) \* N, pour j\_ieme aller-retour

ou

N + 2N + 4 N + 8N + … + 2^(j-1)\*N = M

~~Suivant ce pattern on peut dire pour xième voyage, le nombre d’octets envoyés est : 2∧(x-1) \* N~~

~~Le nombre d’aller-retour qui doit écoulés avant que notre entité ne puisse envoyer M octets est : X tel que 2∧(x-1) \* N >= M.~~

**Exercice 7** (7 points)

**Les réseaux organisés en datagrammes routent chaque paquet à part, indépendamment de tous les autres. Les réseaux à circuit virtuel n’ont pas à faire ce travail pour chaque paquet puisque tous les paquets suivent un chemin prédéfini. Est-ce que cela signifie que les réseaux à circuit virtuel n’ont pas besoin d`être capable de router les paquets isolés qui proviennent d’une station arbitraire et qui vont vers une autre station arbitraire ? Expliquer votre réponse.**

Les réseaux à circuit virtuel sont capables de router tout comme les réseaux orientes non-connexion, mais c’est une décision de routage qui se fait seulement à la configuration, mais pas dans le cadre normal de fonctionnement de tels réseaux.

**Exercice 8** (10 points)

**Deux entités de transports communiquent à travers un réseau fiable. Si on considère que le temps pour transmettre un segment est de 1 unité, que le temps de propagation de bout en bout est de 3 unités et qu’il faut 2 unités pour transmettre les données d’un segment reçu à l’utilisateur de la couche transport. L’émetteur obtient initialement un crédit de 7 segments. Le récepteur lui alloue des crédits à chaque fois qu’il le peut. Quel est le débit maximum qu’on peut atteindre ?**

Il faut 1 unité pour transmettre un segment + 3 unité pour qu’il se propage jusqu’au destinataire + 2 unités pour que le destinataire aie accès à la couche transport. En admettant qu’on envoie les 7 segments disponibles d’un coup et que l’on rend 7 crédits d’un coup, il faut (1\*7 + 3 + 2) + 6 = 18 unités pour pouvoir renvoyer 7 segments, ce qui nous donne un débit de 18/7 segments/unité.

**Exercice 9** (6 points)

**Supposons que 2 entités (p.ex., ordinateurs) sont connectées via un lien de 100 Mbps et que le temps aller-retour est 1 ms. Calculer la taille minimale de la fenêtre TCP pour réaliser le plus grand débit entre les entités.**

Dans TCP on envoie des segments et chaque segment contient des octets ; Avec un lien de 100 Mb/s on peut envoyer 12 500 000 octets/s.

1octets ---------> 8bits

X octets ----------> 100 000 000 bits

========> X = 12 500 000 octets

Le lien nous permet d’envoyer 12 500 000 octets /s On a un temps aller-retour : 1ms ;

Même si le lien nous permet d’envoyer 12 500 000 octets en une seconde nous on veut savoir combien d'octets on peut envoyer en 1ms ??

12 500 000 octets ------------> 1s

X octets ------------> 0.001s

X = 0.001 \* 12500000 = 12 500 octets

La taille des fenêtres doit donc être de 12 500 octets si on veut réaliser le plus grand débit entre les entités.

**Exercice 10** (5 points)

**Est-ce que les algorithmes de Dijkstra and Bellman-Ford produisent tout le temps les mêmes solutions ? expliquer.**

Non, pas dans tous les cas de figure. Dijkstra donne un résultat erroné si le graphe contient des coûts négatifs. Quant à Bellman-Ford, le résultat est toujours sans erreur pour des coûts positifs ou négatifs. Aussi, les deux algorithmes peuvent être non-déterministes dans le cas qu’il existerait plusieurs chemins avec le même coût il se peut que chaque algorithme donne un chemin différent à chaque exécution.