# Composants intégrés d'un système sur puce

*Pourquoi votre téléphone, plus puissant que les ordinateurs de bureau d'il y a quinze ans, n'a-t-il besoin d'aucun ventilateur ?*

1 · FICHE DE SYNTHÈSE

## À compléter au fil de la séance

### Définition

Un système sur puce (SoC) est un circuit ..... qui regroupe des fonctions autrefois réparties sur ..... assemblés sur une carte.

### Les blocs d'un SoC

| Bloc | Rôle |
|---|---|
| ..... | Exécute les instructions des programmes. |
| GPU | ..... |
| ..... ..... | Communications sans fil (Wi-Fi, 4G/5G, Bluetooth). |
| Gestion d'énergie | ..... |
| ..... | Relie les blocs entre eux, à des débits adaptés. |

### Avantages et limites

- Avantages : ..... , ..... , ..... , ..... .
- Limites : difficile à ..... , impossible à faire ..... .

### Repères

- 1971 : l'Intel 4004, premier ..... , environ ..... transistors.
- Loi de Moore : le nombre de transistors ..... environ tous les ..... .
- 2019 : l'A13 Bionic, environ ..... de transistors.

2 · EXERCICES

## Les exercices à rédiger

*Les numéros sont ceux du site. Les exercices d'application directe et les défis bonus sont en ligne.*

### Exercice 2 · La loi de Moore à l'épreuve Guidé

Le 4004 (1971) comptait environ 2 300 transistors. On suppose que le nombre de transistors double exactement tous les deux ans.

**a)** Combien de doublements y a-t-il entre 1971 et 2019 ?

**b)** Quel nombre de transistors la loi de Moore prévoit-elle pour 2019 ? Donnez un ordre de grandeur.

**c)** Comparez avec les 8,5 milliards de l'A13 Bionic. Proposez une explication à l'écart.

## Exercice 3 · Intégration et miniaturisation Type bac

Répondez en une ou deux phrases à chaque question.

**a)** Citez un avantage à avoir plusieurs cœurs sur une même puce.

**b)** Pourquoi un SoC relie-t-il ses blocs par des bus de débits différents ?

**c)** Quel est l'avantage d'un circuit de petite taille ?

**d)** Citez un inconvénient d'une miniaturisation poussée.

## Exercice 4 · Pas de ventilateur Autonome

Expliquez en 3 ou 4 phrases pourquoi un smartphone n'a pas besoin de ventilateur, contrairement à un ordinateur de bureau performant. Revenez à votre hypothèse du début de séance : était-elle juste ?

3 · SUPPORTS À DÉCOUPER

## Le jeu des étiquettes

*Découpez les sept étiquettes, puis placez-les sur les zones du schéma.*




✂

| | | |
|---|---|---|
| **CPU** | **GPU** | **Accélérateur d'IA** |
| **Gestion d'énergie** | **Enclave sécurisée** | **Contrôleur mémoire** |



# Processus et ordonnancement

*Votre ordinateur a quelques cœurs et fait tourner des centaines de programmes. Qui décide, à chaque instant, lequel a le droit de s'exécuter ?*

1 · FICHE DE SYNTHÈSE

## À compléter au fil de la séance

### Programme et processus

Un programme est un fichier ..... . Un processus est un programme ..... .

Chaque processus a un numéro unique, le ..... . Le numéro de son parent s'appelle le ..... . Les processus forment un ..... .

| Commande | Effet |
|---|---|
| `ps -ef` | ..... |
| ..... | termine un processus désigné par son PID |

### Les états d'un processus

| État | Signification |
|---|---|
| ..... | Attend son tour pour utiliser le processeur. |
| Élu | ..... |
| ..... | Attend un événement extérieur (donnée du disque…). |

Vigilance : un processus bloqué qui obtient sa ressource repasse à l'état ..... , jamais directement ..... .

### L'ordonnancement

| Politique | Principe | Point faible |
|---|---|---|
| FIFO | ..... ..... | ..... ..... ..... |
| Plus court d'abord | ..... ..... | Risque de ..... pour les longs. |
| Tourniquet | Chacun reçoit un ..... , puis retourne en fin de ..... . | Changements de processus coûteux. |

2 · EXERCICES

## Les exercices à rédiger

*Les numéros sont ceux du site. Les exercices d'application directe et les défis bonus sont en ligne.*

### Exercice 7 · Un tourniquet à la main Guidé

Trois processus arrivent à l'instant 0, dans cet ordre : P1 (durée 6), P2 (durée 4), P3 (durée 2). Le quantum vaut 2.

**a)** Dessinez le diagramme de Gantt de l'ordonnancement par tourniquet.

| 0–1 | 1–2 | 2–3 | 3–4 | 4–5 | 5–6 | 6–7 | 7–8 | 8–9 | 9–10 | 10–11 | 11–12 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | | | | | | | |

**b)** À quel instant chaque processus se termine-t-il ?

### Exercice 8 · FIFO contre plus court d'abord Guidé

Les mêmes processus arrivent tous à l'instant 0 : P1 (6), P2 (4), P3 (2). Le **temps d'attente** d'un processus est le temps écoulé avant son premier passage sur le processeur.

**a)** Calculez le temps d'attente de chaque processus avec FIFO (ordre P1, P2, P3), puis le temps d'attente moyen.

**b)** Même question avec « plus court d'abord ».

**c)** Quelle politique est la plus avantageuse ici ?

## Exercice 9 · Lire la sortie de ps Type bac

On a obtenu l'affichage suivant :

```
UID    PID  PPID  CMD
root     1     0  /sbin/init
eleve  812     1  /usr/bin/bash
eleve  930   812  python3 jeu.py
eleve  931   812  firefox
eleve  944   931  firefox --onglet
```

**a)** Quel est le PID du processus parent de `python3 jeu.py` ? Quel programme est-ce ?

**b)** Dessinez l'arbre des processus.

**c)** Le programme `jeu.py` ne répond plus. Quelle commande permet de le terminer ?

## Exercice 11 · Trois processus, un seul fichier Maison · machine

Que se passe-t-il quand plusieurs processus écrivent **en même temps** dans le **même fichier** ? Cet exercice se fait sur machine, en classe ou à la maison.

*Si vous ne pouvez pas exécuter les programmes, ou si vous êtes bloqué, des résultats réels sont fournis à la fin de l'énoncé : ils suffisent pour faire tout l'exercice.*

**Préparation.** Créez un dossier `concurrence` et enregistrez-y les trois programmes suivants.

`ecritfichier.py` écrit 1 000 lignes de la forme `0012345 : 0`, `0012345 : 1`…, où `0012345` est son PID :

```
import time
from os import getpid

pid = str(getpid()).zfill(7)            # PID complété par des zéros
with open("test.txt", "w") as fichier:
    time.sleep(1 - time.time() % 1)     # départ groupé
    for i in range(1000):
        fichier.write(pid + " : " + str(i) + "\n")
        fichier.flush()                 # écriture immédiate
        time.sleep(0.001)               # petite pause
```

`lanceur.py` démarre trois copies de `ecritfichier.py` **en même temps** :

*Les programmes* `lanceur.py` *et* `analyse.py` *sont à copier depuis le site (exercice 11 de la séance B.1).*

`analyse.py` lit `test.txt` et résume son contenu :

**TROIS DÉTAILS DU PROGRAMME**

`zfill(7)` complète le PID par des zéros à gauche : toutes les lignes ont ainsi exactement la même longueur.

`time.sleep(1 - time.time() % 1)` fait attendre chaque copie jusqu'à la prochaine seconde pleine : les trois démarrent ensemble, comme au coup de pistolet. Sans cette ligne, chaque copie aurait fini avant que la suivante ait démarré, et on ne verrait rien.

`fichier.flush()` force l'écriture immédiate dans le fichier. Sans elle, Python accumule le texte en mémoire et ne l'écrit que par gros paquets.

**Partie A · Observer.** Exécutez `lanceur.py` (dans VS Code : bouton ▶ en haut à droite), attendez la fin (de 1 à 15 secondes selon l'ordinateur), puis exécutez `analyse.py`.

**a)** Combien de lignes le fichier `test.txt` contient-il ? Combien de lignes les trois processus ont-ils écrites au total, d'après le programme ?

**b)** Quels PID apparaissent dans le fichier, et sur combien de lignes chacun ?

**c)** La ligne n° k contient-elle toujours le nombre k ? (Les lignes sont numérotées à partir de 0.)

**d)** Relancez le tout. Obtenez-vous exactement le même résultat ?

**e)** Proposez une hypothèse : que sont devenues les lignes manquantes ?

**Partie B · Changer une seule lettre.** Dans `ecritfichier.py`, remplacez `open("test.txt", "w")` par `open("test.txt", "a")`. Supprimez `test.txt`, puis relancez `lanceur.py` et `analyse.py`.

**a)** Combien de lignes obtenez-vous maintenant ? Combien par PID ?

**b)** La ligne n° k contient-elle encore le nombre k ? Comment les lignes des trois processus sont-elles disposées ?

**c)** Cherchez ce que signifient les modes `"w"` et `"a"` de la fonction `open`. Votre hypothèse de la partie A se confirme-t-elle ?

**Partie C · Sur papier.** Chaque processus garde en mémoire son propre **curseur** : la position où il écrira son prochain caractère. Le curseur part de 0 à l'ouverture du fichier et avance du nombre de caractères écrits. On suppose que chaque ligne occupe 10 caractères, et que l'ordonnanceur fait se succéder les écritures dans cet ordre :

| Étape | Processus | Écrit sa ligne n° |
|---|---|---|
| 1 | P1 | 0 |
| 2 | P1 | 1 |

| Étape | Processus | Écrit sa ligne n° |
|---|---|---|
| 3 | P2 | 0 |
| 4 | P2 | 1 |
| 5 | P2 | 2 |
| 6 | P3 | 0 |
| 7 | P1 | 2 |

**a)** En mode `"w"`, à quelle position (en caractères) chaque écriture a-t-elle lieu ?

**b)** Quel est le contenu final du fichier, ligne par ligne ?

**c)** En mode `"a"`, chaque écriture a lieu à la fin du fichier. Quel est alors le contenu final ?

**Partie D · Un fichier chacun.** Remplacez maintenant `open("test.txt", "a")` par `open("test_" + pid + ".txt", "w")`, puis relancez `lanceur.py`.

**a)** Avant d'exécuter : combien de fichiers allez-vous obtenir, et que contiendra chacun ?

**b)** Vérifiez. Pourquoi le problème de la partie A a-t-il disparu ?

**c)** On veut pourtant garder un seul fichier commun, ouvert en mode `"w"`. Quelle règle faudrait-il imposer aux trois processus ?

**ET SOUS LINUX ?**

Sous Linux, inutile d'écrire `lanceur.py` : le terminal lance les trois copies en une seule ligne, grâce au symbole `&`, qui envoie chaque commande en arrière-plan et rend la main immédiatement : `python3 ecritfichier.py & python3 ecritfichier.py & python3 ecritfichier.py &`

Le terminal affiche aussitôt les trois PID, puis « Done » quand chaque copie se termine. On analyse ensuite le fichier avec des commandes, par exemple `wc -l test.txt` pour compter les lignes. C'est la version proposée dans la séance B.1 bis, sur clé USB Linux.

**Pourquoi pas `&` sous Windows ?** Dans l'invite de commandes de Windows (`cmd`), le symbole `&` existe aussi, mais il signifie « puis » : les trois copies s'exécuteraient l'une après l'autre, sans aucune concurrence, et un seul PID resterait dans le fichier. D'où le `lanceur.py`, qui fonctionne sur tous les systèmes.

> **Une différence pour la partie B.** Sous Linux, le mode `"a"` garantit que chaque écriture va à la fin du fichier, même quand plusieurs processus écrivent en même temps. Sous Windows, Python se place à la fin puis écrit, en deux temps : très rarement, une ligne peut se perdre. Si vous obtenez un peu moins de 3 000 lignes, c'est l'explication… et c'est encore le même phénomène.

---

### Pas d'ordinateur, ou bloqué ? Voici des résultats réels

Ces résultats ont été obtenus en exécutant réellement les programmes. Sous Windows, les PID sont plus grands, mais le comportement est le même.

**Partie A, mode `"w"`** — sortie du lancement puis de l'analyse :

```
Trois processus lancés, PID : [352, 353, 354]
Les trois copies sont terminées.

Nombre de lignes : 1000
Lignes écrites par chaque PID : {'0000354': 292, '0000353': 142,
'0000352': 566}
Lignes n° k qui contiennent le nombre k : 1000
Lignes abîmées : 0
Lignes n° 500 à 504 : ['0000352 : 500', '0000352 : 501', '0000352 :
502', '0000352 : 503', '0000352 : 504']
```

Extrait de `test.txt`, lignes n° 106 à 112 :

```
0000354 : 106
0000354 : 107
0000354 : 108
0000353 : 109
0000353 : 110
0000353 : 111
0000353 : 112
```

Un second lancement, identique, a donné :

```
Lignes écrites par chaque PID : {'0000291': 911, '0000292': 88,
'0000290': 1}
```

**Partie B, mode `"a"`** :

```
Trois processus lancés, PID : [300, 301, 302]
Les trois copies sont terminées.

Nombre de lignes : 3000
Lignes écrites par chaque PID : {'0000302': 1000, '0000300': 1000,
'0000301': 1000}
Lignes n° k qui contiennent le nombre k : 1
Lignes abîmées : 0
Lignes n° 500 à 504 : ['0000300 : 167', '0000302 : 166', '0000301 :
167', '0000300 : 168', '0000302 : 167']
```

Extrait de `test.txt`, lignes n° 600 à 606 :

```
0000302 : 199
0000301 : 200
0000300 : 201
0000302 : 200
0000301 : 201
0000300 : 202
0000302 : 201
```

**Partie D, un fichier par processus** :

```
test_0000310.txt : 1000 lignes
test_0000311.txt : 1000 lignes
test_0000312.txt : 1000 lignes
```

> **SUR MACHINE**
> L'exercice 10 (programmer un tourniquet) se fait sur machine, à partir du site. Pour l'exercice 11, les programmes peuvent être copiés depuis le site.

3 · SUPPORTS À DÉCOUPER

## Jouez l'ordonnanceur

*Découpez les cinq cartes processus. Chaque volontaire trace un bâton par unité de temps passée avec le stylo « processeur ».*

| | |
|---|---|
| **Navigateur**<br>Durée : 6 unités de temps<br>Bâtons : | **Musique**<br>Durée : 2 unités de temps<br>Bâtons : |
| **Messagerie**<br>Durée : 3 unités de temps<br>Bâtons : | **Antivirus**<br>Durée : 8 unités de temps<br>Bâtons : |
| **Horloge**<br>Durée : 1 unité de temps<br>Bâtons : | |

✂

### Relevé du secrétaire

*Notez dans chaque case le processus qui tient le stylo pendant cette unité de temps.*

**Manche 1 · Premier arrivé, premier servi**

| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | | | | | | | | | | |

**Manche 2 · Tourniquet (quantum = 2)**

| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | | | | | | | | | | | |

**Manche 3 · Plus court d'abord**

| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | | | | | | | | | | | |

# L'interblocage

*Quatre voitures arrivent en même temps à un carrefour sans feux, et chacune laisse la priorité à celle de droite. Combien de temps vont-elles attendre ?*

## 1 · FICHE DE SYNTHÈSE

## À compléter au fil de la séance

### Définition

Un interblocage est une situation où plusieurs processus ..... pour obtenir des ..... , si bien qu'aucun ne peut ..... .

### Le graphe d'attente

- Flèche processus → ressource : « ..... ».
- Flèche ressource → processus : « ..... ».

À retenir : il y a interblocage si et seulement si le graphe contient un ..... .

### Éviter ou résoudre

| Stratégie | Principe |
|---|---|
| ..... | Tous les processus demandent leurs ressources ..... . |
| Tout ou rien | ..... ..... |
| ..... ..... | On cherche un cycle et on arrête un processus impliqué. |

Ne pas confondre : un processus bloqué attend une ressource qui qui ..... ; dans un interblocage, elle ..... .

## 2 · EXERCICES

### Exercice 14 · Les deux imprimantes Guidé

Deux programmes A et B doivent chacun imprimer un document en recto sur l'imprimante 1 et en verso sur l'imprimante 2.

**a)** Décrivez un scénario qui conduit à un interblocage.

**b)** Dessinez le graphe d'attente correspondant.

**c)** Proposez une règle qui rend l'interblocage impossible.

## Exercice 15 · Trois processus, trois ressources Type bac

On observe la situation suivante :

| Processus | Détient | Attend |
|---|---|---|
| P1 | R1 | R2 |
| P2 | R2 | R3 |
| P3 | R3 | R1 |

**a)** Dessinez le graphe d'attente.

**b)** Les processus sont-ils en interblocage ? Justifiez.

**c)** P3 libère R3 sans rien attendre d'autre. La situation se débloque-t-elle ? Dans quel ordre les processus peuvent-ils terminer ?

## Exercice 16 · Le philosophe gaucher Guidé

Au dîner des philosophes, chacun prend d'abord sa fourchette gauche, puis la droite. Le philosophe n° 5 décide de faire l'inverse : il prend d'abord sa fourchette droite.

L'interblocage où chacun tient sa fourchette gauche est-il encore possible ? Justifiez.

> **SUR MACHINE**
> L'exercice 17 (suivre la chaîne des attentes) se fait sur machine, à partir du site.

3 · SUPPORTS À DÉCOUPER

### Les quatre stylos

*Recopier les 4 étiquettes et collez-en une sur chaque stylo pour la manche 2.*

| Stylo n° 1 | Stylo n° 2 | Stylo n° 3 | Stylo n° 4 |
|---|---|---|---|

# Le protocole de routage RIP

*Un message envoyé au Japon traverse une quinzaine de routeurs, dont certains au fond de l'océan. Aucun d'eux ne connaît le chemin complet. Comment arrive-t-il à destination ?*

1 · FICHE DE SYNTHÈSE

## À compléter au fil de la séance

### Routeur et table de routage

Un routeur achemine les paquets ..... . Sa table indique, pour chaque destination, la ..... : le voisin à qui transmettre le paquet.

### Le protocole RIP

- Chaque routeur envoie sa table à ses ..... , toutes les 30 secondes.
- Il retient la route qui compte le moins de ..... .
- Distance maximale : ..... sauts ; 16 signifie « ..... ».
- RIP est un protocole à ..... : un routeur ne connaît jamais la carte complète.

Quand plus aucune table ne change, le réseau a ..... .

Limite : RIP ne tient pas compte du ..... des liaisons.

2 · EXERCICES

## Les exercices à rédiger

*Les numéros sont ceux du site. Les exercices d'application directe et les défis bonus sont en ligne.*

### Exercice 19 · La table du routeur A Guidé

Complétez la table de routage RIP du routeur A, une fois le réseau convergé.

| Destination | Passerelle | Distance |
|---|---|---|
| B | | |
| C | | |
| D | | |
| E | | |
| F | | |

### Exercice 20 · Convergence pas à pas Guidé

Quatre routeurs sont reliés en ligne : W — X — Y — Z.

**a)** Écrivez la table de W au départ (il ne connaît que ses voisins).

**b)** Écrivez la table de W après le premier échange avec X.

**c)** Écrivez la table de W après le deuxième échange. Le réseau a-t-il convergé ?

## Exercice 21 · Une liaison tombe en panne Type bac

**a)** Quel chemin RIP choisit-il de B vers F ?

**b)** La liaison B—E tombe en panne. Quel chemin est désormais utilisé de B vers F ? Le nombre de sauts change-t-il ?

## Exercice 22 · La table du routeur C Autonome

Établissez la table de routage RIP complète du routeur C.

| Destination | Passerelle | Distance |
|---|---|---|
| | | |
| | | |
| | | |
| | | |
| | | |

## Exercice 23 · La limite de RIP Type bac

Sur le réseau de l'entreprise, la liaison A—F est un vieux câble Ethernet très lent ; toutes les autres sont rapides.

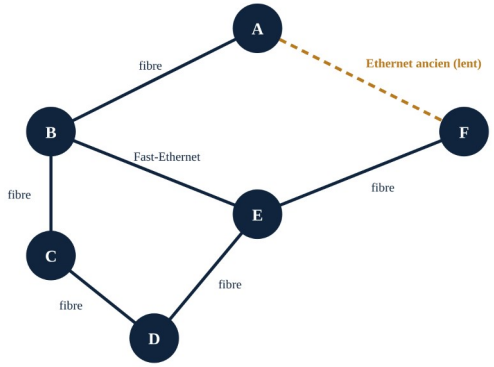
**a)** Quel chemin RIP choisit-il de A vers F ?

**b)** Expliquez pourquoi ce choix pose problème.

3 · SUPPORTS À DÉCOUPER

## La classe devient un réseau

*Une fiche par routeur. Au départ, n'inscrivez que vos voisins directs, à distance 1.*



## Routeur A

Voisins directs : .....

| Destination | Passerelle | Distance |
|---|---|---|
| | | |
| | | |
| | | |
| | | |
| | | |

✂ – – –

# Le protocole de routage RIP

*Un message envoyé au Japon traverse une quinzaine de routeurs, dont certains au fond de l'océan. Aucun d'eux ne connaît le chemin complet. Comment arrive-t-il à destination ?*

## 1 · FICHE DE SYNTHÈSE

### Routeur et table de routage

Un routeur achemine les paquets ..... . Sa table indique, pour chaque destination, la ..... : le voisin à qui transmettre le paquet.

### Le protocole RIP

- Chaque routeur envoie sa table à ses ..... , toutes les 30 secondes. Il retient la route qui compte le moins de ..... .
- Distance maximale : ..... sauts ; 16 signifie « ..... ».
- RIP est un protocole à ..... : un routeur ne connaît jamais la carte complète.

Quand plus aucune table ne change, le réseau a ..... .

Limite : RIP ne tient pas compte du ..... des liaisons.

### Exemple : la table du routeur D, échange après échange

On reprend le réseau de l'activité (routeurs A à F). Au démarrage, la fibre **D–E est en maintenance** : D n'a donc qu'un seul voisin, C. Elle est rebranchée juste avant le 3e échange. Les échanges sont simultanés : à chaque échange, chaque routeur envoie à ses voisins la table qu'il avait à la fin de l'échange précédent.

*Complétez la table de D après chaque échange, en suivant l'exemple projeté. Dans la colonne Changement, notez « nouveau » ou « plus court ».*

**Au départ**

| Destination | Passerelle | Distance | Changement |
|---|---|---|---|
| C | C | 1 | |

**Après l'échange 1**

| Destination | Passerelle | Distance | Changement |
|---|---|---|---|
| | | | |
| | | | |

**Après l'échange 2**

| Destination | Passerelle | Distance | Changement |
|---|---|---|---|
| | | | |
| | | | |
| | | | |
| | | | |

**Après l'échange 3 (la fibre D–E est rebranchée)**

| Destination | Passerelle | Distance | Changement |
|---|---|---|---|
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |

Une route n'est remplacée que si la nouvelle est ..... plus courte. À égalité, le routeur garde ..... .

2 · EXERCICES

## Les exercices à rédiger

*Les numéros sont ceux du site. Les exercices d'application directe et les défis bonus sont en ligne.*

### Exercice 19 · La table du routeur A Guidé

Complétez la table de routage RIP du routeur A, une fois le réseau convergé.

| Destination | Passerelle | Distance |
|---|---|---|
| B | | |
| C | | |
| D | | |
| E | | |
| F | | |

### Exercice 20 · Convergence pas à pas Guidé

Quatre routeurs sont reliés en ligne : W — X — Y — Z.

**a)** Écrivez la table de W au départ (il ne connaît que ses voisins).

**b)** Écrivez la table de W après le premier échange avec X.

**c)** Écrivez la table de W après le deuxième échange. Le réseau a-t-il convergé ?

### Exercice 21 · Une liaison tombe en panne Type bac

**a)** Quel chemin RIP choisit-il de B vers F ?

**)** La liaison B—E tombe en panne. Quel chemin est désormais utilisé de B vers F ? Le nombre de sauts change-t-il ?

### Exercice 22 · La table du routeur C Autonome

Établissez la table de routage RIP complète du routeur C.

| Destination | Passerelle | Distance |
|---|---|---|
| | | |
| | | |
| | | |
| | | |
| | | |

### Exercice 23 · La limite de RIP Type bac

Sur le réseau de l'entreprise, la liaison A—F est un vieux câble Ethernet très lent ; toutes les autres sont rapides.

**a)** Quel chemin RIP choisit-il de A vers F ?

**b)** Expliquez pourquoi ce choix pose problème.



# Le protocole de routage OSPF

*Votre GPS vous propose deux itinéraires : le plus court en kilomètres et le plus rapide. Lequel prenez-vous ? Les routeurs d'Internet se posent la même question.*

1 · FICHE DE SYNTHÈSE

## À compléter au fil de la séance

### Le protocole OSPF

- Chaque routeur connaît ..... du réseau.
- Il minimise ..... des liaisons.
- OSPF est un protocole à ..... .

Coût d'une liaison de débit d (en ..... ) : `coût =`

.....

Vigilance : 100 Mbit/s = ..... bit/s et 1 Gbit/s = ..... bit/s.

| | RIP | OSPF |
|---|---|---|
| Ce qu'il minimise | ..... | ..... |
| Ce que connaît un routeur | ..... ..... | ..... |
| Prise en compte du débit | ..... | ..... |

L'algorithme utilisé par OSPF pour trouver le chemin de coût minimal s'appelle l'algorithme de ..... .

## Exercice 26 · OSPF contre RIP, de A vers F Guidé

**a)** Calculez le coût total de chacun des trois chemins : A → F, A → B → E → F, A → B → C → D → E → F.

**b)** Quel chemin OSPF choisit-il ?

**c)** Comparez avec le choix de RIP.

## Exercice 27 · Panne de la liaison B—E Type bac

La liaison B—E tombe en panne.

**a)** Quel est le nouveau chemin de coût minimal entre A et F ?

**b)** Combien de sauts compte-t-il ? Qu'en pensez-vous ?

## Exercice 28 · De C vers F Type bac

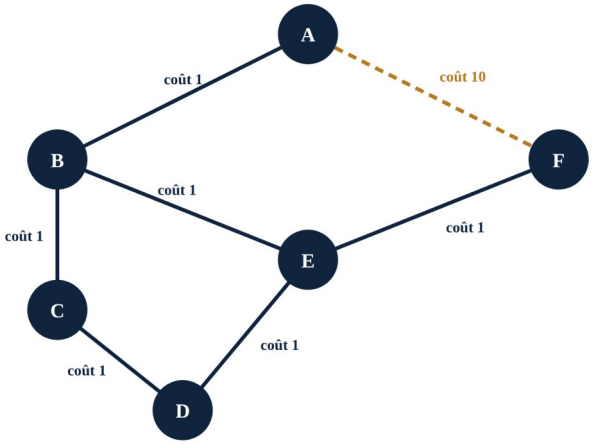
Déterminez le chemin de coût minimal de C vers F. Y en a-t-il plusieurs ?

## Exercice 30 · Pourquoi OSPF dans les vrais réseaux ? Autonome

Expliquez en quelques phrases pourquoi un réseau d'entreprise qui mélange fibre, Wi-Fi et vieux câbles utilise plutôt OSPF que RIP.

## Chacun a la carte

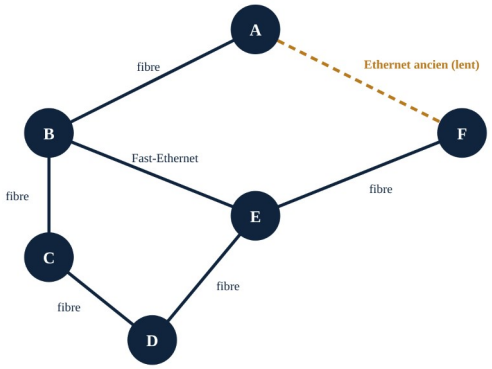
*Une carte par binôme.*



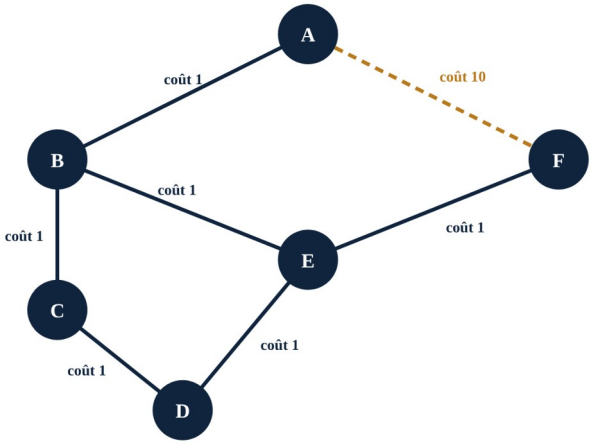

3 · SUPPORTS À DÉCOUPER

## La classe devient un réseau

*Une fiche par routeur. Au départ, n'inscrivez que vos voisins directs, à distance 1.*







## Routeur A

Voisins directs : .....

| Destination | Passerelle | Distance |
|---|---|---|
| | | |
| | | |
| | | |
| | | |
| | | |

✂ – – –

# Chiffrement symétrique et asymétrique

*Comment partager un secret avec quelqu'un que vous n'avez jamais rencontré, alors que tout le monde peut lire ce que vous lui envoyez ?*

1 · FICHE DE SYNTHÈSE

## À compléter au fil de la séance

### Chiffrement symétrique

La ..... sert à chiffrer et à déchiffrer. Avantage : ..... . Point faible : ..... .

XOR : deux bits égaux donnent ..... , deux bits différents donnent ..... . Pour déchiffrer, on refait un XOR avec ..... .

Une clé de 8 bits n'offre que ..... possibilités : une attaque par ..... la casse immédiatement.

### Chiffrement asymétrique

- Chaque personne possède une ..... , connue de tous, et une ..... , secrète.
- Pour écrire en secret à Alice, on chiffre avec la clé ..... ; elle déchiffre avec sa clé ..... .
- Point faible : ..... que le symétrique.

Analogie : la clé publique est un ..... que tout le monde peut fermer, mais que seul son propriétaire peut ouvrir.

2 · EXERCICES

## Les exercices à rédiger

*Les numéros sont ceux du site. Les exercices d'application directe et les défis bonus sont en ligne.*

### Exercice 33 · Qui utilise quelle clé ? Guidé

B veut envoyer un message secret m à A, en chiffrement asymétrique. Ève espionne le réseau.

**a)** Quelle clé B utilise-t-il pour chiffrer m ?

**b)** Quelle clé A utilise-t-il pour déchiffrer ?

**c)** Ève connaît la clé publique de A. Peut-elle déchiffrer le message ?

### Exercice 36 · Symétrique ou asymétrique ? Type bac

**a)** Pourquoi n'utilise-t-on pas le chiffrement asymétrique pour chiffrer de gros volumes de données ?

**b)** Alice veut envoyer à Bob, et à lui seul, un message secret, en chiffrement asymétrique. Décrivez les étapes.

### Exercice 37 · Le meilleur des deux mondes Autonome

Le symétrique est rapide mais pose le problème de l'échange de clé ; l'asymétrique règle ce problème mais il est lent. Imaginez une méthode qui combine les deux.

**SUR MACHINE**

Les exercices 34 (César) et 35 (casser une clé de 8 bits) se font sur machine, à partir du site.

# HTTPS et le chiffrement RSA

*Votre navigateur et le serveur de votre banque ne se sont jamais rencontrés. Pourtant, en quelques millisecondes, ils se mettent d'accord sur une clé secrète que personne d'autre ne connaît. Comment ?*

1 · FICHE DE SYNTHÈSE

## À compléter au fil de la séance

### HTTPS

HTTPS = HTTP + ..... . L'asymétrique sert seulement à ..... ; ensuite, toute la communication est chiffrée en ..... .

- Le serveur envoie son ..... , qui contient sa clé publique.
- Le client génère une clé symétrique, la chiffre avec ..... et l'envoie.
- Le serveur la déchiffre avec ..... .

Un certificat est signé par une ..... : il prouve que la clé publique appartient bien au site.

### Approfondissement : RSA et Diffie-Hellman

- RSA : $N = p \times q$ ; chiffrement $c =$ ..... ; déchiffrement $m =$ ..... .
- Sa sécurité repose sur la difficulté de ..... N.
- Diffie-Hellman : Alice et Bob fabriquent une clé commune ..... .

## Exercice 39 · Remettre la négociation dans l'ordre Guidé

Remettez ces étapes dans l'ordre :

- (1) Le serveur déchiffre la clé symétrique avec sa clé privée.
- (2) Le client vérifie le certificat et génère une clé symétrique.
- (3) Le client demande une connexion sécurisée.
- (4) Les échanges sont chiffrés en symétrique.
- (5) Le serveur envoie son certificat.
- (6) Le client envoie la clé symétrique, chiffrée avec la clé publique du serveur.

## Exercice 40 · Diffie-Hellman avec des nombres Guidé

Alice et Bob choisissent en public $p = 23$ et $g = 5$. Alice choisit en secret $a = 6$, Bob $b = 15$.

**a)** Alice calcule $A = g^a \mod p$ et l'envoie. Que vaut $A$ ?

**b)** Bob calcule $B = g^b \mod p$ et l'envoie. On admet que $B = 19$.

**c)** Alice calcule $B^a \mod p$ et Bob calcule $A^b \mod p$. On admet que les deux valent 2. Que représente ce nombre ?

**d)** Ève a vu passer $p$, $g$, $A$ et $B$. Que lui manque-t-il ?

## Exercice 41 · Un exemple complet de RSA Guidé

On prend $p = 5$, $q = 11$, $e = 23$ et $d = 7$.

**a)** Calculez $N$ et $n$.

**b)** Vérifiez que $e \times d \equiv 1 \pmod{n}$.

**c)** Chiffrez le message $m = 8$.

**d)** Déchiffrez le résultat obtenu.

## Exercice 43 · Pourquoi de si grands nombres ? Type bac

**a)** Pourquoi RSA utilise-t-il, en pratique, des nombres premiers de plusieurs centaines de chiffres ?

**b)** Pourquoi connaître N, qui est public, ne permet-il pas de retrouver la clé privée ?

## Exercice 45 · À quoi sert le certificat ? Autonome

Expliquez à quoi sert un certificat, en plus de transporter la clé publique du serveur. Que pourrait faire un attaquant s'il n'existait pas ?