Chiffrement symétrique

Christophe Viroulaud

Terminale - NSI

Archi 20

La communication sur internet est organisée en couches.

Couche application (Navigateur)
Couche TCP (Transport)
Couche IP (Internet)
Couche réseau (Matérielle)

Tableau 1 – Protocole TCP/IP

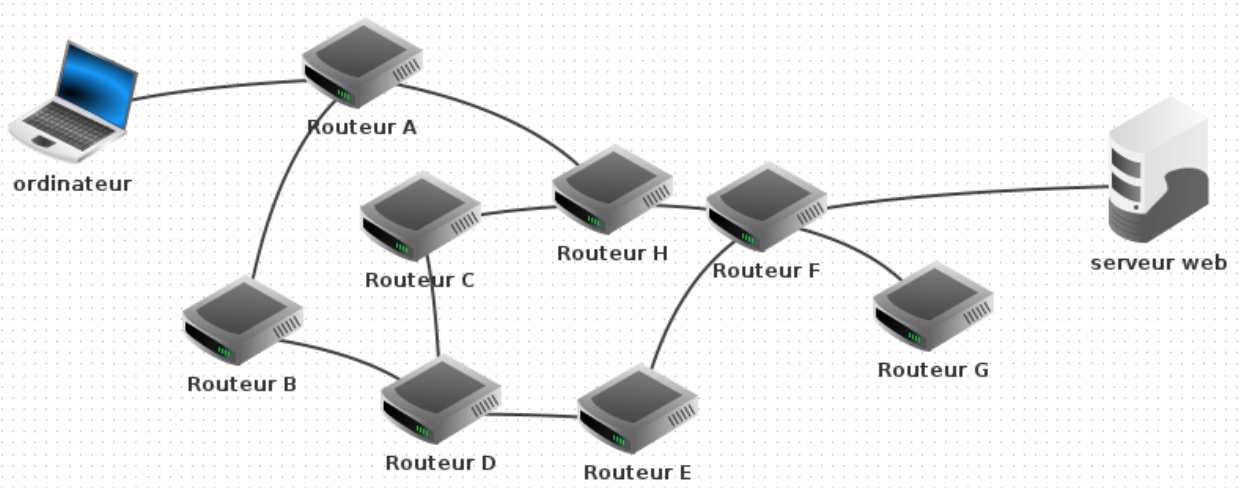


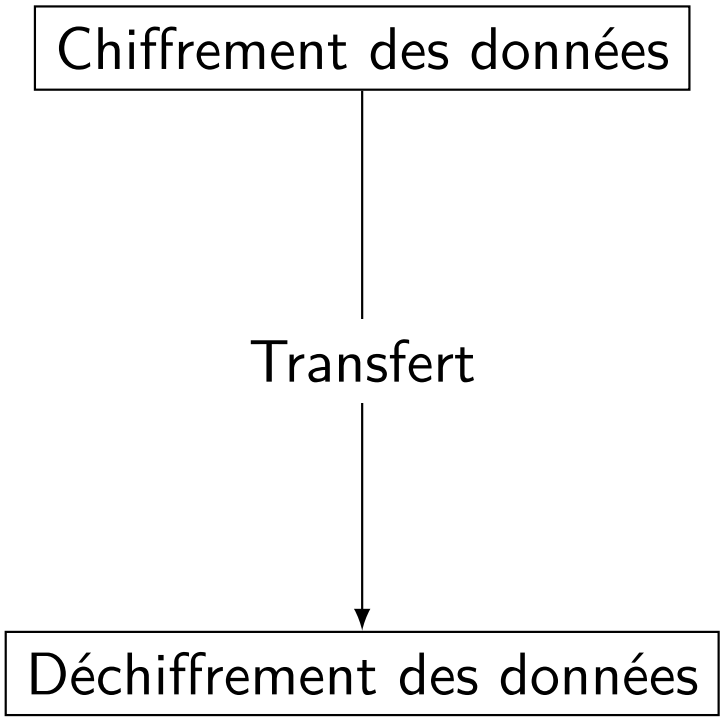
FIGURE 1 – Les paquets IP transitent sur le réseau internet en circulant de routeurs en routeurs.

En théorie, rien n'interdit à un routeur d'inspecter un paquet et donc d'en connaître son contenu.

Comment chiffrer le contenu des communications ?

Sommaire

Chiffrement symétrique - Principe



- La source utilise une *fonction de chiffrement* pour coder un message `m` avec une clé de chiffrement `k`. La fonction produit en sortie un message chiffré `s`.

$$\texttt{chiffrement(m, k)} \rightarrow \texttt{s}$$

- La source utilise une *fonction de chiffrement* pour coder un message `m` avec une clé de chiffrement `k`. La fonction produit en sortie un message chiffré `s`.

$$\texttt{chiffrement(m, k)} \rightarrow \texttt{s}$$

- Le destinataire utilise une *fonction de déchiffrement* pour décoder le message `s` avec la clé de chiffrement `k`. La fonction produit en sortie le message clair `m`.

$$\texttt{déchiffrement(s, k)} \rightarrow \texttt{m}$$

À retenir

Dans un chiffrement symétrique on utilise la même clé pour chiffrer et déchiffrer le message.

Sommaire

Chiffrement de César

Le chiffrement de César utilise un décalage alphabétique comme clé de chiffrement. Par exemple, avec la clé **+2** :

- A devient C
- B devient D
- ...
- Z devient B

Activité 1 : Écrire la fonction `chiffrement(message: str, cle: int)` $\rightarrow$ `str` qui code `message`.
On n'utilisera que des caractères majuscules ASCII dans le message et on supprimera les espaces. Dans un premier temps, on ne s'occupera pas du *débordement de l'alphabet*. Ainsi l'appel

```
>>> chiffrement("Z", 1)
```

renverra le caractère `[` situé à la 91° position du code ASCII.

Correction


```
1  def chiffrement(message: str, cle: int) -> str:
2      sortie = ""
3      for lettre in message:
4          # code ASCII de la lettre chiffrée
5          code = ord(lettre) + cle
6          # ajout
7          sortie = sortie+chr(code)
8      return sortie
```

Activité 2 : Modifier la fonction pour que l'appel

```
>>> chiffrement("Z", 1)
```

renvoie la lettre **A**

Correction

```
1  def chiffrement(message: str, cle: int) -> str:
2      sortie = ""
3      for lettre in message:
4          # code ASCII de la lettre chiffrée
5          code = (ord(lettre) + cle) % 91  # Z = 90
6          # ajustement du code ASCII
7          if code < ord("A"):
8              code = code+ord("A")
9          # ajout
10         sortie = sortie+chr(code)
11     return sortie
```

Activité 3 : Écrire la fonction `dechiffrement(message: str, cle: int)` $\rightarrow$ `str` qui déchiffre `message` en prenant en compte le *débordement de l'alphabet*.

Correction