Introduction aux réseaux

Valeur d'accueil et de reconversion en informatique (VARI1)

Daniel Porumbel (dp.cnam@gmail.com)

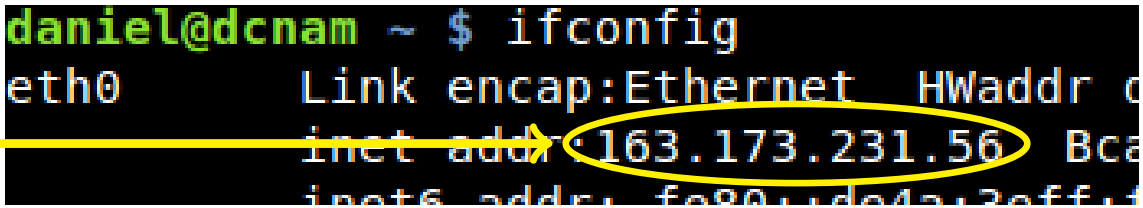
http://cedric.cnam.fr/~porumbed/vari1/

Plan

Les adresses IP

Toute machine connectée possède une adresse IP

- exemples : 10.0.0.1, 163.173.0.1, ou 202.2.1.15
- C'est comme une plaque d'immatriculation,
  - sauf qu'on peut avoir plusieurs IPs si on a plusieurs cartes réseaux
- `ifconfig` : afficher notre adresse IP

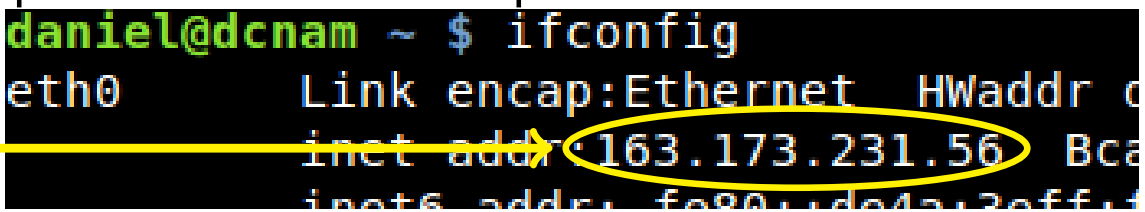


- `route -n` ⟹ afficher l'IP de la passerelle par défaut
  - ex., une box ADSL qui nous connecte à l'Internet
- `ping 163.173.228.2` ⟹ tester si la machine d'adresse IP `163.173.228.2` est active
- `traceroute google.fr` ⟹ voir les machines/routeurs qui nous relient à la machine `google.fr` (en Californie)
  - Un message est transféré de routeur en routeur (une dizaine de routeurs sont traversés) pour l'acheminer à `google.fr`

Les adresses IP

Toute machine connectée possède une adresse IP

- exemples : 10.0.0.1, 163.173.0.1, ou 202.2.1.15
- C'est comme une plaque d'immatriculation,
  - sauf qu'on peut avoir plusieurs IPs si on a plusieurs cartes réseaux
- `ifconfig` : afficher notre adresse IP



- `route -n` $\Longrightarrow$ afficher l'IP de la passerelle par défaut
  - ex., une box ADSL qui nous connecte à l'Internet
- `ping 163.173.228.2` $\Longrightarrow$ tester si la machine d'adresse IP `163.173.228.2` est active
- `traceroute google.fr` $\Longrightarrow$ voir les machines/routeurs qui nous relient à la machine `google.fr`[1] (en Californie)
  - Un message est transféré de routeur en routeur (une dizaine de routeurs sont traversés) pour l'acheminer à `google.fr`

1. Utiliser `www.iplocation.net` pour voir la location géographique des routeurs.

Commandes Réseau Linux

La commande `ifconfig` affiche ou modifie l'IP

- `sudo ifconfig eth0 10.0.0.1/8` $\Longrightarrow$ nouvelle IP [2]
  - Si le réseau utilise un service/serveur DHCP (Dynamic Host Configuration Protocol), l'IP est configuré automatiquement (ou avec `dhclient`), et non pas avec `ifconfig`
- La machine a aussi une adresse de diffusion (en. : *broadcast*) qui désigne toutes les machines du (sous-)réseau

`ssh SERVEUR` : connexion au `SERVEUR` en mode text

- L'option `-X` de `ssh` permet d'afficher les commandes graphiques sur le serveur d'affichage (X) de la machine locale
- lancer `xeyes` après `export DISPLAY=:0.0` en boucle `for`

2. Il faut parfois taper "`stop network-manager`" pour arrêter le gestionnaire graphique de réseaux, sinon il peut annuler l'effet d'`ifconfig`.

Commandes Réseau Linux

La commande `ifconfig` affiche ou modifie l'IP

- `sudo ifconfig eth0 10.0.0.1/8` $\Longrightarrow$ nouvelle IP [2]
  - Si le réseau utilise un service/serveur DHCP (Dynamic Host Configuration Protocol), l'IP est configuré automatiquement (ou avec `dhclient`), et non pas avec `ifconfig`
- La machine a aussi une adresse de diffusion (en. : *broadcast*) qui désigne toutes les machines du (sous-)réseau

`ssh SERVEUR` : connexion au `SERVEUR` en mode text

- L'option $-X$ de `ssh` permet d'afficher les commandes graphiques sur le serveur d'affichage (X) de la machine locale
- lancer `xeyes` après `export DISPLAY=:0.0` en boucle `for`

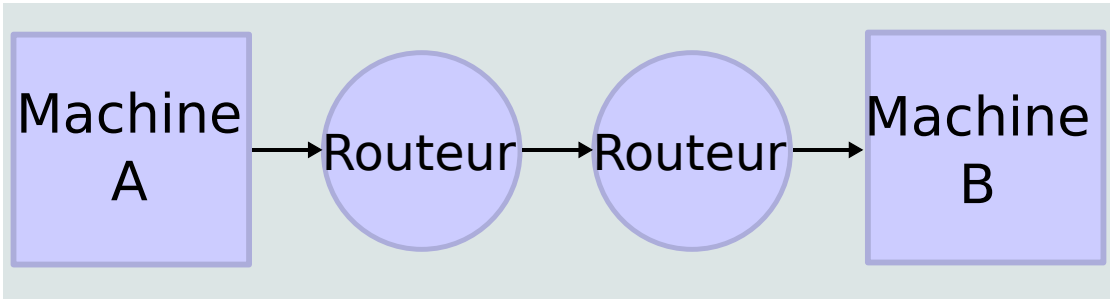
2. Il faut parfois taper "`stop network-manager`" pour arrêter le gestionnaire graphique de réseaux, sinon il peut annuler l'effet d'`ifconfig`.

La pile TCP/IP

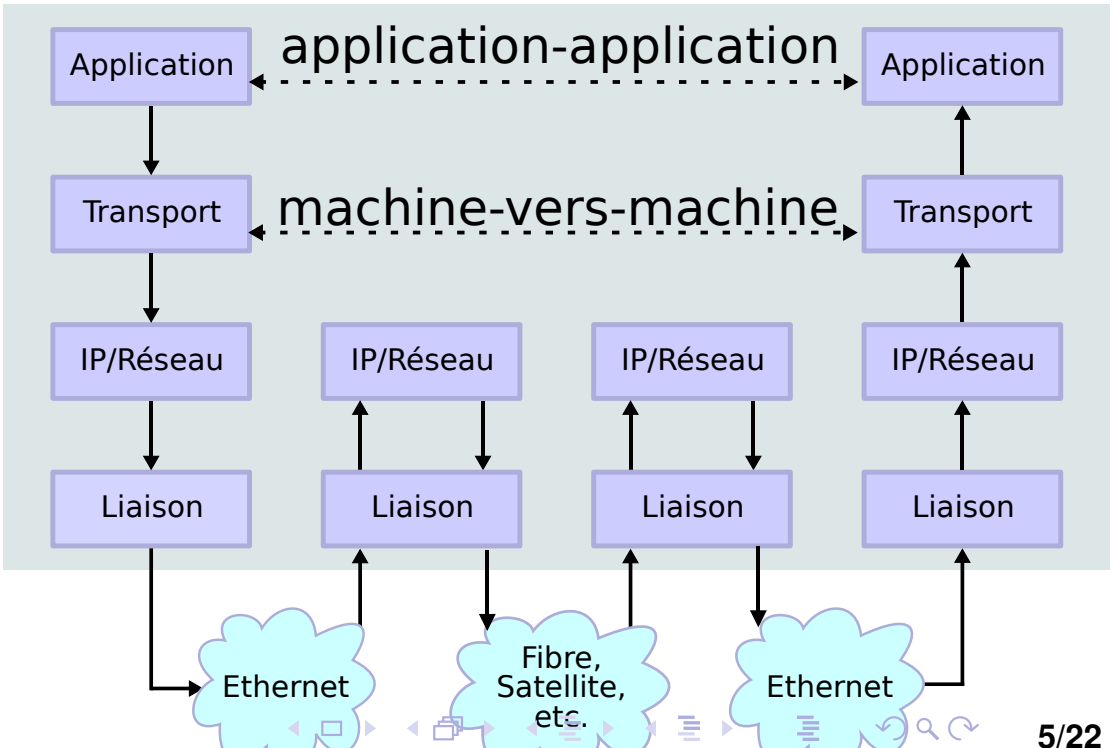
Quatre couches de protocoles qui s'appuient sur la couche physique :

1. Couche Applications (navigateurs Web, ssh, FTP)
2. Couche Transport (TCP/UDP)
3. Couche IP/Réseaux
4. Couche Liaison

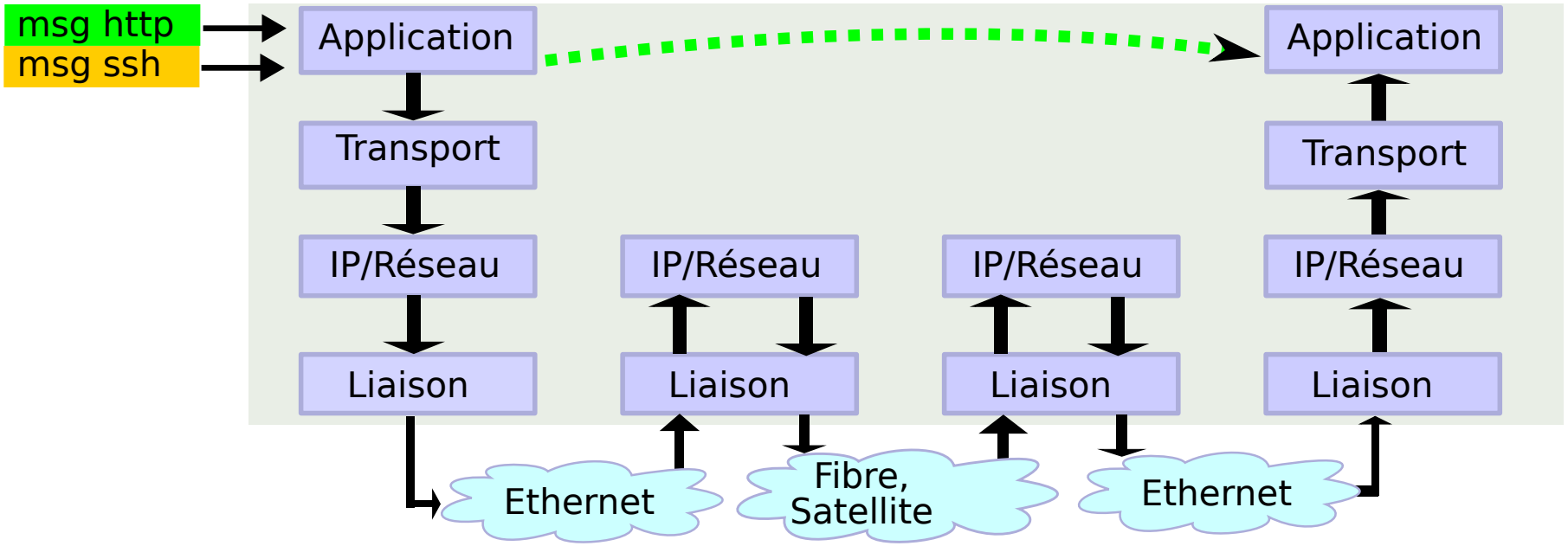
Topologie Réseau



Flux de données



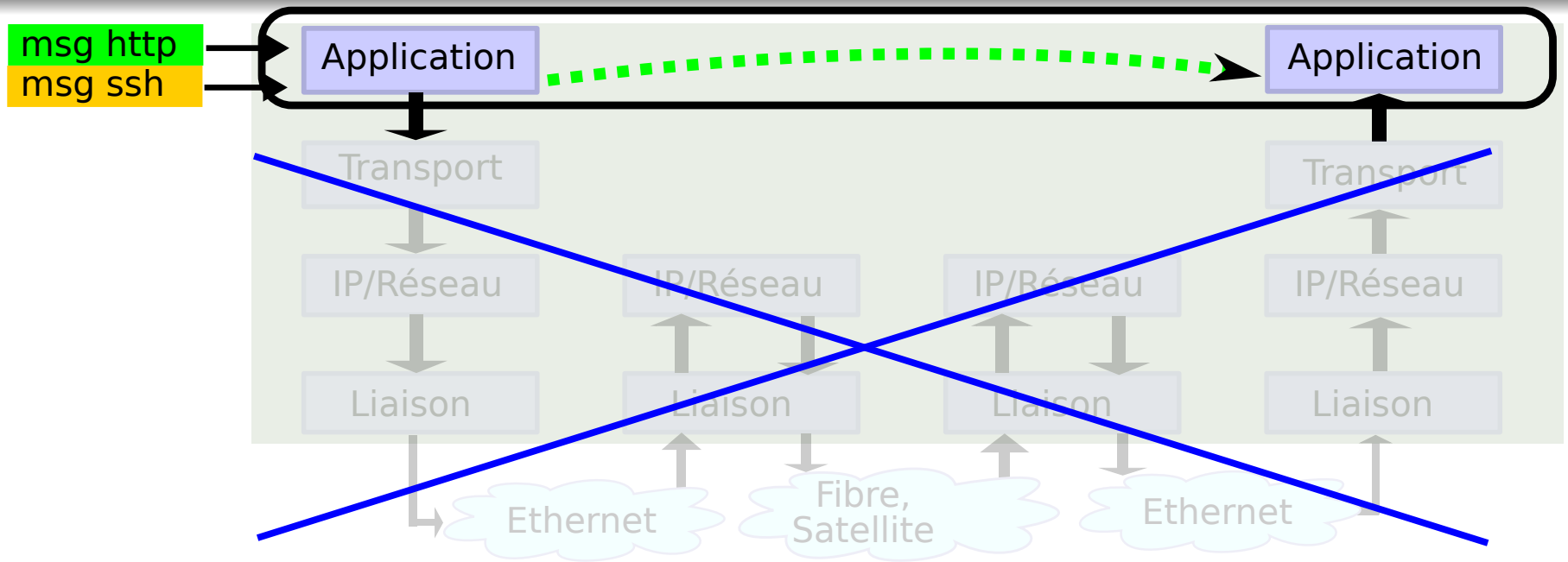
La Couche Applications



À la couche Applications on s'intéressent aux :

- programmes pour l'utilisateur (navigateur web, ssh)
- la cryptographie
- protocoles dits de haut niveau (`http`, `https`, `ftp`, `telnet`)
  - On ne s'intéressent pas aux routeurs et passerelles qui relient les machines, ni au câbles/connecteurs physiques : ces aspects sont la tache des couches inférieures

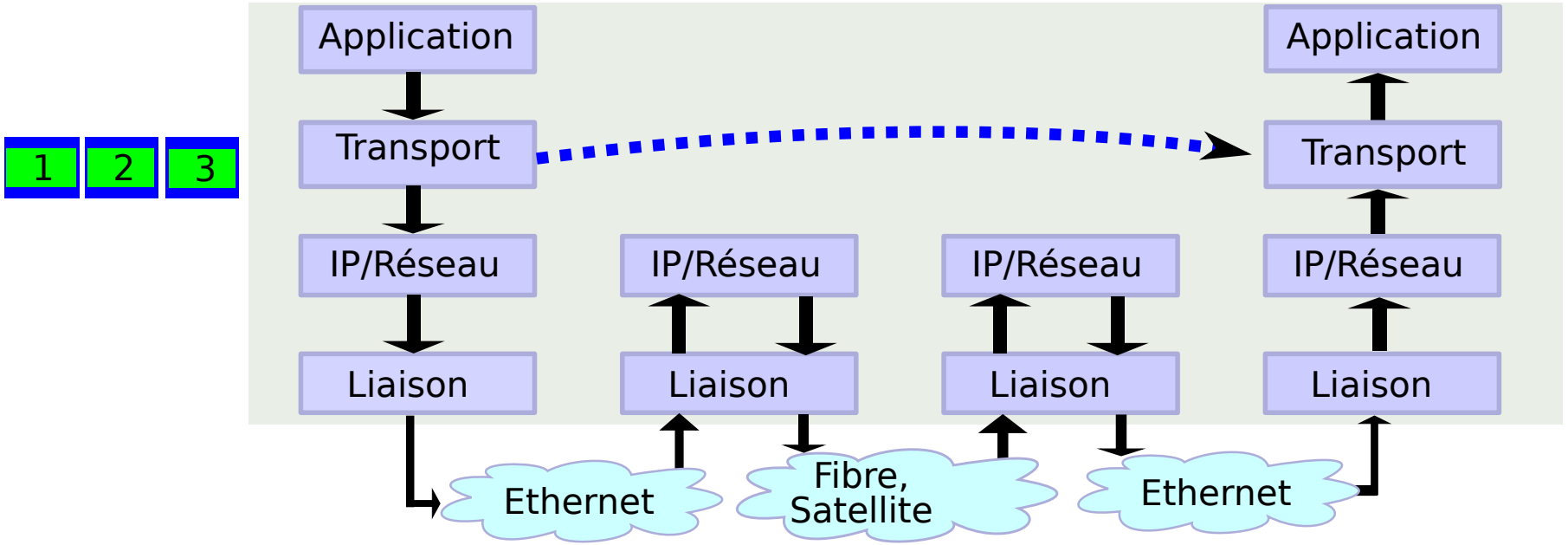
La Couche Applications



À la couche Applications on s'intéressent aux :

- programmes pour l'utilisateur (navigateur web, ssh)
- la cryptographie
- protocoles dits de haut niveau (`http`, `https`, `ftp`, `telnet`)
  - On ne s'intéressent pas aux routeurs et passerelles qui relient les machines, ni au câbles/connecteurs physiques : ces aspects sont la tache des couches inférieures

La couche Transport

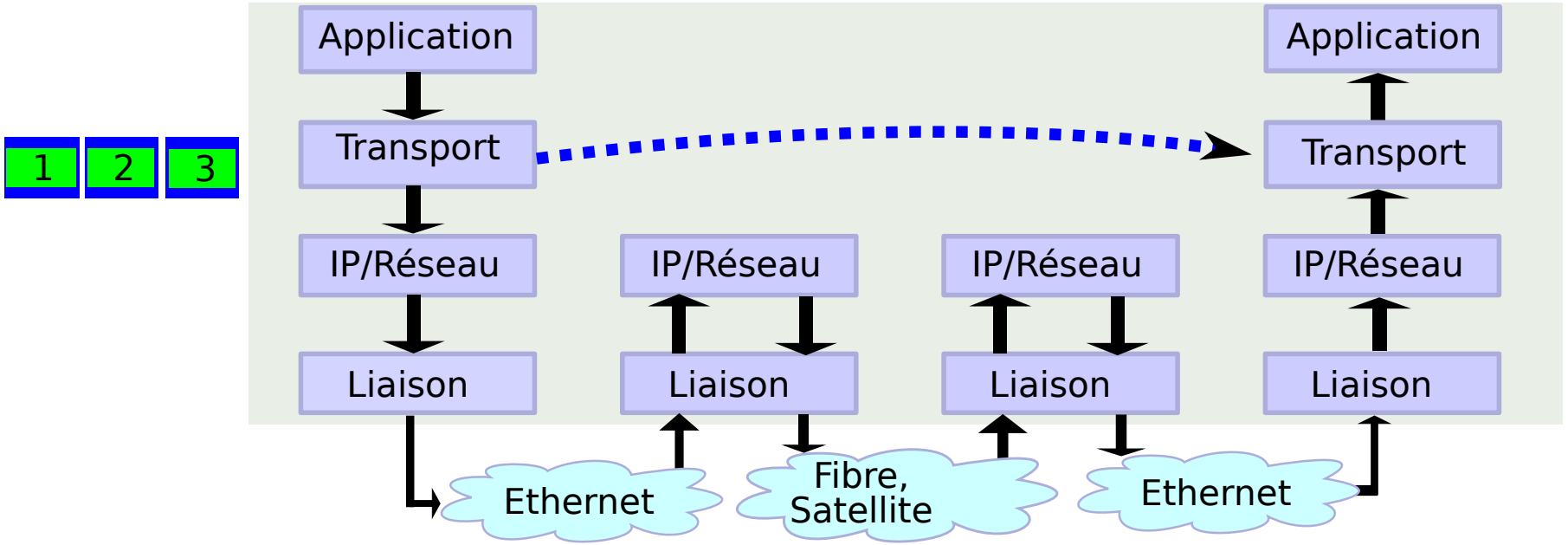


- Le *message* de la couche Application est découpé en *segments* de type TCP (avec connexion) ou UDP (sans connexion)
- La communication utilise des sockets
  - une socket = une machine(IP)+ un port, ex. `10.0.0.1:80`
    - un port ≈ case courrier dans une institution (machine)
  - Exemple de connexion à `cedric.cnam.fr:80` via `netcat`

```
netcat cedric.cnam.fr 80<<<"get /"
```

- La transmission TCP ou UDP ne s'intéresse pas aux routeurs qui assurent la liaison entre les deux machines

La couche Transport

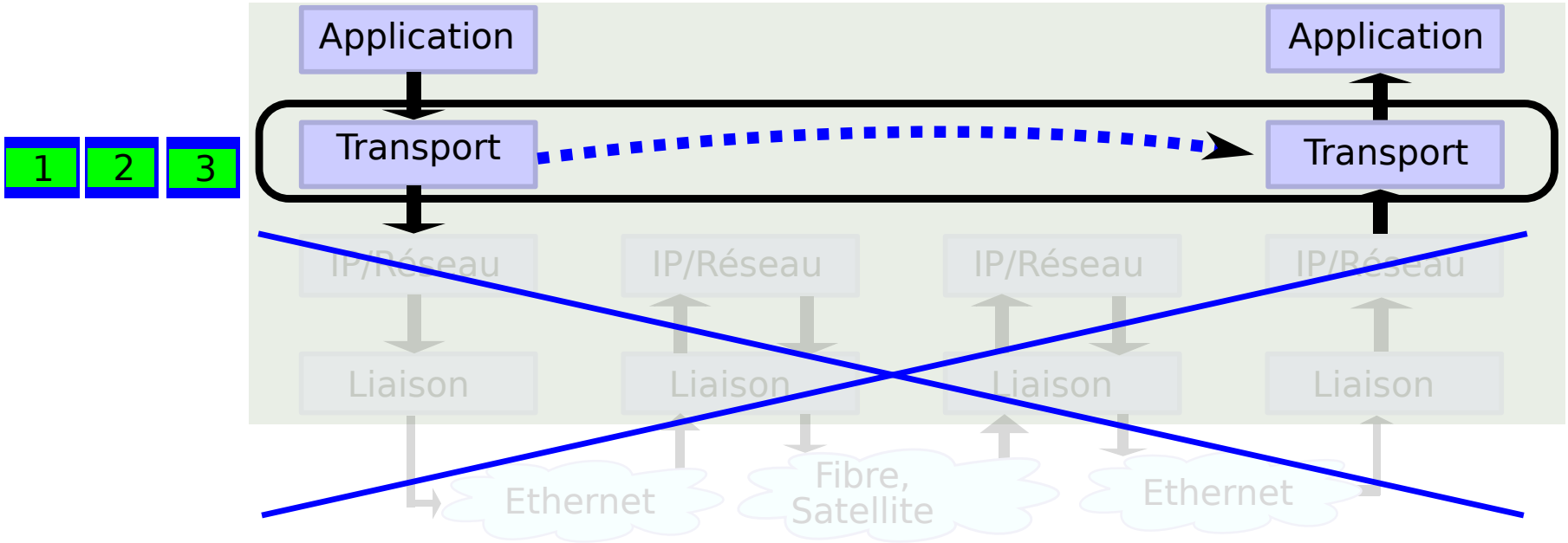


- Le *message* de la couche Application est découpé en *segments* de type TCP (avec connexion) ou UDP (sans connexion)
- La communication utilise des sockets
  - une socket = une machine(IP)+ un port, ex. `10.0.0.1:80`
    - un port ≈ case courrier dans une institution (machine)
  - Exemple de connexion à `cedric.cnam.fr:80` via `netcat`

```
netcat  cedric.cnam.fr  80<<<"get  /"
```

- La transmission TCP ou UDP ne s'intéresse pas aux routeurs qui assurent la liaison entre les deux machines

La couche Transport

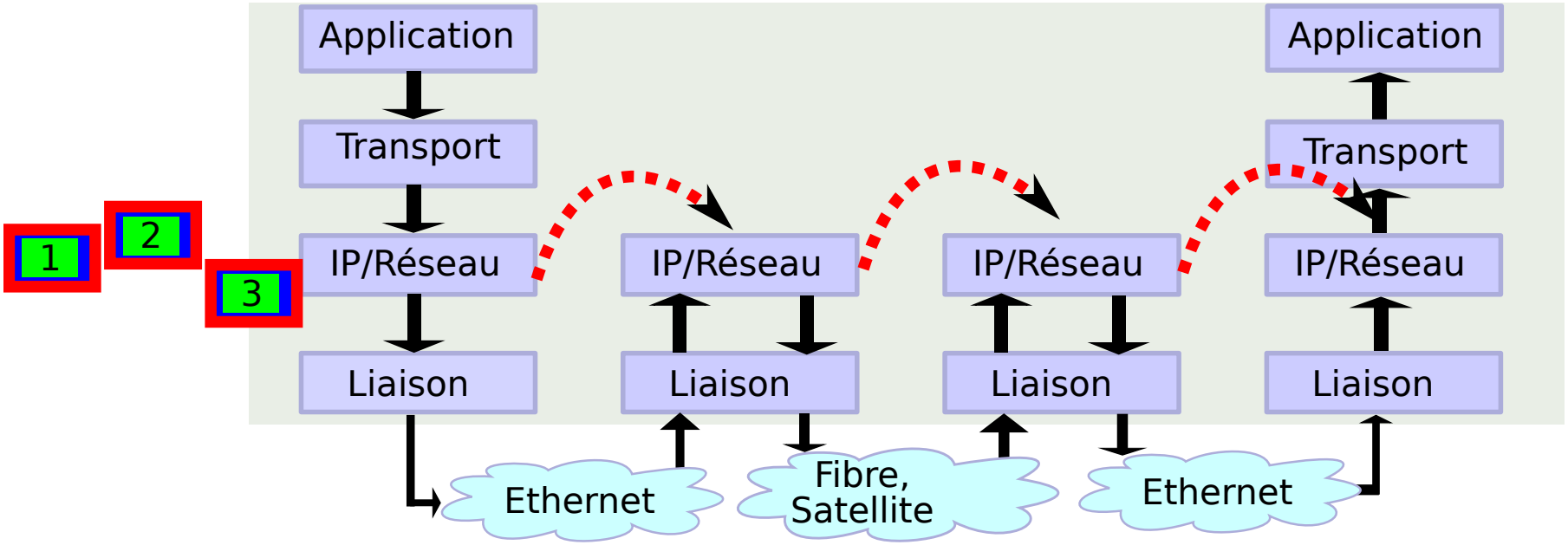


- Le *message* de la couche Application est découpé en *segments* de type TCP (avec connexion) ou UDP (sans connexion)
- La communication utilise des sockets
  - une socket = une machine(IP)+ un port, ex. `10.0.0.1:80`
    - un port ≈ case courrier dans une institution (machine)
  - Exemple de connexion à `cedric.cnam.fr:80` via `netcat`

```
netcat  cedric.cnam.fr  80<<<"get  /"
```

- La transmission TCP ou UDP ne s'intéresse pas aux routeurs qui assurent la liaison entre les deux machines

La couche Réseaux

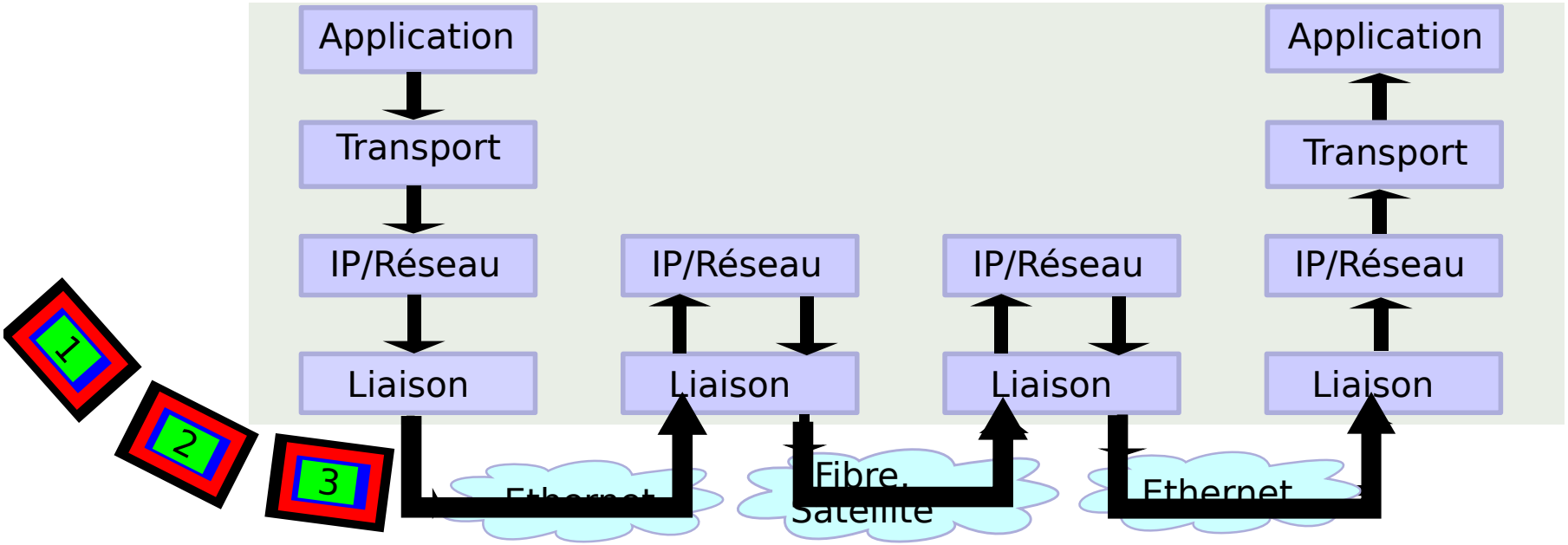


- Couche transport : segment envoyé de A vers B

$$\Longrightarrow$$

- Couche réseau : transfert $A \to A_1 \to A_2 \to A_3 \ldots\ldots\cdots \to B$
  - Chaque routeur peut “cacher” un pirate qui écoute la communication
- La couche *réseau* (ou couche IP) gère la circulation des *paquets* à travers le réseau en assurant leur routage.

La couche Liaison



- La couche *liaison* indique comment les paquets sont transportés sur la couche physique (Ethernet/Wifi)
- L'en-tête des trames Ethernet comporte l'adresse MAC destination (MAC=adresse physique carte réseau)
- Le protocole ARP fait la connexion IP $\rightarrow$ MAC.
  - Etant donnée une adresse IP de notre sous-réseau, quelle est son adresse MAC ? La commande `arp -n` affiche les MACs connus par notre machine

Exemple d'utilisation couche Applications

Pour aller sur `www.cnam.fr` :

1. Le navigateur web demande au service DNS (Domain Name Server) l'adresse IP du serveur `www.cnam.fr`
   - C'est comme si on tapait `dig www.cnam.fr`
2. Le DNS retourne `163.173.128.40` par exemple.
3. Le navigateur envoie une requête HTTP à cette adresse IP
4. Le serveur web répond avec un fichier `html` (⊕ des scripts `javascript` éventuellement)

- La communication passe par une connexion TCP sur le port 80 de `www.cnam.fr` (port supérieur à 1024 pour la machine locale)

Le site peut demander de déposer un cookie qui stocke une valeur dans le navigateur. Cette valeur peut être demandée plus tard au navigateur, ex, pour faire du pistage (publicitaire).

Exemple d'utilisation couche Applications

Pour aller sur `www.cnam.fr` :

1. Le navigateur web demande au service DNS (Domain Name Server) l'adresse IP du serveur `www.cnam.fr`
   - C'est comme si on tapait `dig www.cnam.fr`
2. Le DNS retourne `163.173.128.40` par exemple.
3. Le navigateur envoie une requête HTTP à cette adresse IP
4. Le serveur web répond avec un fichier `html` (⊕ des scripts `javascript` éventuellement)
   - La communication passe par une connexion TCP sur le port 80 de `www.cnam.fr` (port supérieur à 1024 pour la machine locale)

Le site peut demander de déposer un cookie qui stocke une valeur dans le navigateur. Cette valeur peut être demandée plus tard au navigateur, ex, pour faire du pistage (publicitaire).

Exemple d'utilisation couche Applications

Pour aller sur `www.cnam.fr` :

1. Le navigateur web demande au service DNS (Domain Name Server) l'adresse IP du serveur `www.cnam.fr`
   - C'est comme si on tapait `dig www.cnam.fr`
2. Le DNS retourne `163.173.128.40` par exemple.
3. Le navigateur envoie une requête HTTP à cette adresse IP
4. Le serveur web répond avec un fichier `html` (⊕ des scripts `javascript` éventuellement)
   - La communication passe par une connexion TCP sur le port 80 de `www.cnam.fr` (port supérieur à 1024 pour la machine locale)

Le site peut demander de déposer un cookie qui stocke une valeur dans le navigateur. Cette valeur peut être demandée plus tard au navigateur, ex, pour faire du pistage (publicitaire).

Exemple d'utilisation couche Applications

Pour aller sur `www.cnam.fr` :

1. Le navigateur web demande au service DNS (Domain Name Server) l'adresse IP du serveur `www.cnam.fr`
   - C'est comme si on tapait `dig www.cnam.fr`
2. Le DNS retourne `163.173.128.40` par exemple.
3. Le navigateur envoie une requête HTTP à cette adresse IP
4. Le serveur web répond avec un fichier `html` ($\oplus$ des scripts `javascript` éventuellement)
   - La communication passe par une connexion TCP sur le port 80 de `www.cnam.fr` (port supérieur à 1024 pour la machine locale)

Le site peut demander de déposer un cookie qui stocke une valeur dans le navigateur. Cette valeur peut être demandée plus tard au navigateur, ex, pour faire du pistage (publicitaire).

Exemple d'utilisation couche Applications

Pour aller sur `www.cnam.fr` :

1. Le navigateur web demande au service DNS (Domain Name Server) l'adresse IP du serveur `www.cnam.fr`
   - C'est comme si on tapait `dig www.cnam.fr`
2. Le DNS retourne `163.173.128.40` par exemple.
3. Le navigateur envoie une requête HTTP à cette adresse IP
4. Le serveur web répond avec un fichier `html` ($\oplus$ des scripts `javascript` éventuellement)
   - La communication passe par une connexion TCP sur le port 80 de `www.cnam.fr` (port supérieur à 1024 pour la machine locale)

Le site peut demander de déposer un cookie qui stocke une valeur dans le navigateur. Cette valeur peut être demandée plus tard au navigateur, ex, pour faire du pistage (publicitaire).

Couche Application : le DNS (*nom* → IP)

Le service DNS est très souvent utilisé pour trouver les adresses IP des machines/sites.

Si on tape `ping vlad.cnam.fr` :

- Le serveur DNS de la machine est interrogé :
  - Voir `/etc/resolv.conf` sous Linux
  - Utiliser `/etc/hosts` pour accélérer la navigation web : plus besoin d'attendre la réponse TCP/UDP du serveur DNS
- Le serveur DNS a deux options :
  1. Il renvoie l'adresse IP s'il a cette information OU
  2. Demande cette information à un serveur DNS de niveau supérieur
- Vulnérabilité : ce processus n'est pas trop sécurisé ⇒ si la réponse DNS est fausse, la navigation Web est compromise
  - à savoir : pas de cryptographie par défaut pour le DNS
  - UDP : pas de cryptographie, mais possible de sécuriser le DNS via TCP

Couche Application : le DNS (*nom* → IP)

Le service DNS est très souvent utilisé pour trouver les adresses IP des machines/sites.

Si on tape `ping vlad.cnam.fr` :

- Le serveur DNS de la machine est interrogé :
  - Voir `/etc/resolv.conf` sous Linux
  - Utiliser `/etc/hosts` pour accélérer la navigation web : plus besoin d'attendre la réponse TCP/UDP du serveur DNS
- Le serveur DNS a deux options :
  1. Il renvoie l'adresse IP s'il a cette information OU
  2. Demande cette information à un serveur DNS de niveau supérieur
- Vulnérabilité : ce processus n'est pas trop sécurisé ⇒ si la réponse DNS est fausse, la navigation Web est compromise
  - à savoir : pas de cryptographie par défaut pour le DNS
  - UDP : pas de cryptographie, mais possible de sécuriser le DNS via TCP

Plan

Le plus connu exemple : Phishing

Attaque facile à base d'ingénierie sociale (*social engineering*)

- Envoyer à la victime une page web/mail identique à celui d'un site de confiance (la banque de la victime) et lui demander des mots de passe
- Cibles populaires : banques, amazon, paypal, ebay
- Leçons :
  - vérifier l'adresse web dans la barre d'adresse du navigateur
  - vérifier que la liaision `https` est correctement établie, c.à.d., pas de problème avec le certificat de sécurité
    - Si le DNS ne fonctionne pas correctement, le `http` peut se tromper d'adresse IP (mais pas le `https` et son certificat)
  - Le protocole de mail (SMTP) ne garantit pas et ne vérifie pas l'adresse email d'un expéditeur
    - pas trop dur d'envoyer un email avec expéditeur `bill.gates@microsoft.com`

Le plus connu exemple : Phishing

Attaque facile à base d'ingénierie sociale (*social engineering*)

- Envoyer à la victime une page web/mail identique à celui d'un site de confiance (la banque de la victime) et lui demander des mots de passe
- Cibles populaires : banques, amazon, paypal, ebay
- Leçons :
  - vérifier l'adresse web dans la barre d'adresse du navigateur
  - vérifier que la liaision `https` est correctement établie, c.à.d., pas de problème avec le certificat de sécurité

    - Si le DNS ne fonctionne pas correctement, le `http` peut se tromper d'adresse IP (mais pas le `https` et son certificat)
  - Le protocole de mail (SMTP) ne garantit pas et ne vérifie pas l'adresse email d'un expéditeur
    - pas trop dur d'envoyer un email avec expéditeur `bill.gates@microsoft.com`

Le plus connu exemple : Phishing

Attaque facile à base d'ingénierie sociale (*social engineering*)

- Envoyer à la victime une page web/mail identique à celui d'un site de confiance (la banque de la victime) et lui demander des mots de passe
- Cibles populaires : banques, amazon, paypal, ebay
- Leçons :
  - vérifier l'adresse web dans la barre d'adresse du navigateur
  - vérifier que la liaision `https` est correctement établie, c.à.d., pas de problème avec le certificat de sécurité

    - Si le DNS ne fonctionne pas correctement, le `http` peut se tromper d'adresse IP (mais pas le `https` et son certificat)
  - Le protocole de mail (SMTP) ne garantit pas et ne vérifie pas l'adresse email d'un expéditeur
    - pas trop dur d'envoyer un email avec expéditeur `bill.gates@microsoft.com`

Attention aux failles de programmation

Les pirates cherchent souvent des failles dans *les implémentations des protocoles*. Exemple *C* :

Rappel :*Linux, Windows et Mac OS* sont écrits en *C*

```c
void main(){
   char chaineCaract[5];   //tableau de 5 char
   gets(chaineCaract);     //lecture chaine
   printf(chaineCaract);   //affichage chaine
}
```

Problème : si on saisit une chaine de plus de 5 caractères

$\Longrightarrow$ la fonction `gets(...)` essaye d'écrire au delà des cases mémoires de `chaineCaract`

$\Longrightarrow$ Erreur *buffer overflow*, dépassement de tampon/mémoire

Ce type de problèmes apparaît très souvent lors de la réception de paquets réseaux

$\Longrightarrow$ au lieu de `gets(...)` on peut avoir une lecture réseau

Attention aux failles de programmation

Les pirates cherchent souvent des failles dans *les implémentations des protocoles*. Exemple *C* :

Rappel : *Linux, Windows et Mac OS* sont écrits en *C*

```c
void main(){
  char chaineCaract[5];   //tableau de 5 char
  gets(chaineCaract);     //lecture chaine
  printf(chaineCaract);   //affichage chaine
}
```

Problème : si on saisit une chaine de plus de 5 caractères

$\Longrightarrow$ la fonction `gets(...)` essaye d'écrire au delà des cases mémoires de `chaineCaract`

$\Longrightarrow$ Erreur *buffer overflow*, dépassement de tampon/mémoire

---

Ce type de problèmes apparaît très souvent lors de la réception de paquets réseaux

- au lieu de `gets(...)` on peut avoir une lecture réseau

Attention aux failles de programmation

Les pirates cherchent souvent des failles dans *les implémentations des protocoles*. Exemple *C* :

Rappel : *Linux, Windows et Mac OS* sont écrits en *C*

```c
void main(){
  char chaineCaract[5];   //tableau de 5 char
  gets(chaineCaract);     //lecture chaine
  printf(chaineCaract);   //affichage chaine
}
```

Problème : si on saisit une chaine de plus de 5 caractères

$\Longrightarrow$ la fonction `gets(...)` essaye d'écrire au delà des cases mémoires de `chaineCaract`

$\Longrightarrow$ Erreur *buffer overflow*, dépassement de tampon/mémoire

Ce type de problèmes apparaît très souvent lors de la réception de paquets réseaux

- au lieu de `gets(...)` on peut avoir une lecture réseau

Exemple : le ping de la mort

- Attaque historique réalisée par un paquet ping malformé
- Un ping a normalement une taille de 56 octets $\rightarrow$ risques dépassement de mémoire pour des paquets plus grands
  - Ce dépassement de mémoire provoquait un crash sur plusieurs OS (Windows/Linux)
- Un simple *ping* pouvait provoquer un crash d'une machine cible (Unix, Linux, MacOS, Windows)

Notions de cryptographie

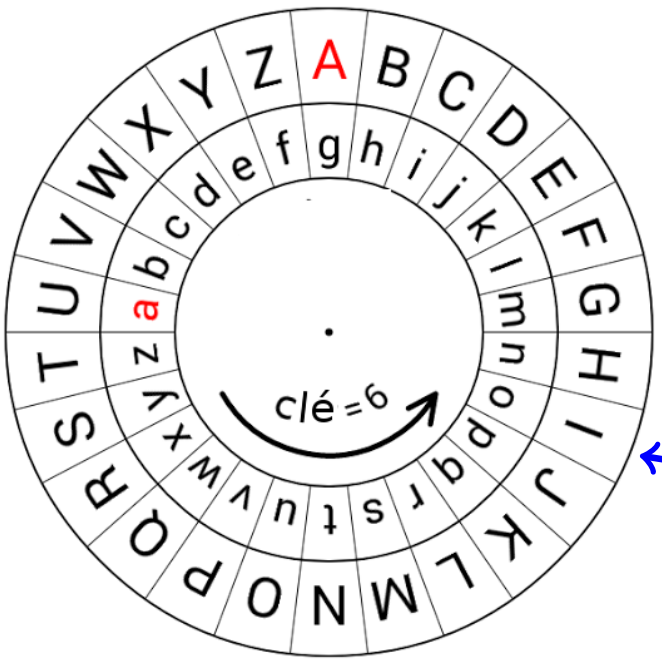
- Objectif de la cryptographie : la confidentialité
  - protéger les messages contre la lecture non autorisée
  - *Exemple :* Une chaîne de télévision payante
- De nombreux problèmes sont évités par cryptographie
  - L'administrateur réseau (ou le fournisseur Internet) ne voit plus les messages que vous échangez

Exemple de code par décalage (César). La clé vaut 6 !

Notions de cryptographie

- Objectif de la cryptographie : la confidentialité
  - protéger les messages contre la lecture non autorisée
  - *Exemple :* Une chaîne de télévision payante
- De nombreux problèmes sont évités par cryptographie
  - L'administrateur réseau (ou le fournisseur Internet) ne voit plus les messages que vous échangez

Exemple de code par décalage (César). La clé vaut 6 !



- Le texte chiffré s'obtient en remplaçant chaque lettre du texte clair original par une lettre à distance fixe (clé de décalage)

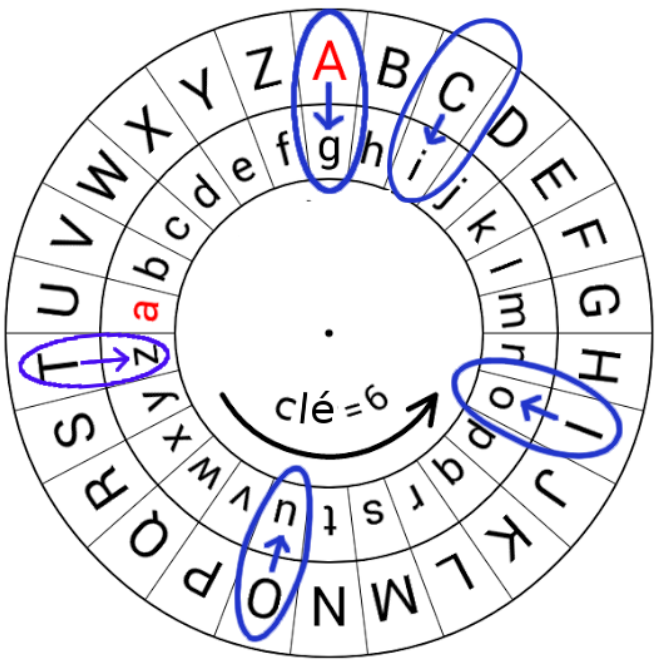
Exemple codage avec clé=6 :

Les lettres de la roue extérieure sont remplacées par celles de la roue intérieure

Notions de cryptographie

- Objectif de la cryptographie : la confidentialité
  - protéger les messages contre la lecture non autorisée
  - *Exemple :* Une chaîne de télévision payante
- De nombreux problèmes sont évités par cryptographie
  - L'administrateur réseau (ou le fournisseur Internet) ne voit plus les messages que vous échangez

Exemple de code par décalage (César). La clé vaut 6 !



- Le texte chiffré s'obtient en remplaçant chaque lettre du texte clair original par une lettre à distance fixe (clé de décalage)

  Exemple codage avec clé=6 :

  `ciao toto→iogu zuzu`

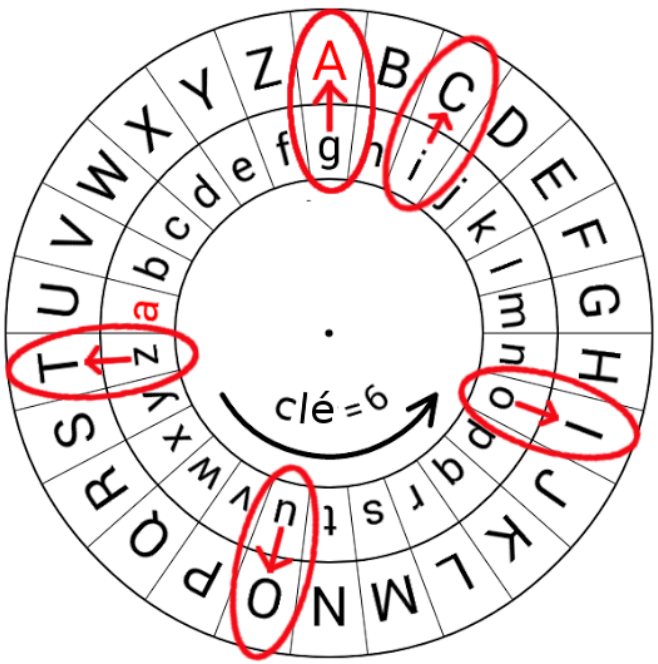
  décodage avec clé=6 :

  `iogu zuzu→ciao toto`

Notions de cryptographie

- Objectif de la cryptographie : la confidentialité
  - protéger les messages contre la lecture non autorisée
  - *Exemple :* Une chaîne de télévision payante
- De nombreux problèmes sont évités par cryptographie
  - L'administrateur réseau (ou le fournisseur Internet) ne voit plus les messages que vous échangez

Exemple de code par décalage (César). La clé vaut 6 !



- Le texte chiffré s'obtient en remplaçant chaque lettre du texte clair original par une lettre à distance fixe (clé de décalage)

Exemple codage avec clé=6 :

`ciao toto→iogu zuzu`

décodage avec clé=6 :

`iogu zuzu→ciao toto`

Cassage du codage César

Soit le message codé :

Uyi n'emqi e jemvi gsrreixvi gi rsqfvi yxmpi eyb wekiw

Deux idées pour décoder :

- Il y a très peu de mots d'une seule lettre en français : a, y
  $\Longrightarrow$ a $\rightarrow$ e (clé=4) ou y $\rightarrow$ e (clé=-20)
- La plus fréquente lettre en français est "e". La plus fréquente lettre du texte est "i", donc e$\rightarrow$i (clé=4)

Avec une clé de 4 on obtient :

Que j'aime à faire connaitre ce nombre utile aux sages

Cassage du codage César

Soit le message codé :

Uyi n'emqi e jemvi gsrreixvi gi rsqfvi yxmpi eyb wekiw

Deux idées pour décoder :

- Il y a très peu de mots d'une seule lettre en français : a, y
  $\Longrightarrow$ a $\rightarrow$ e (clé=4) ou y $\rightarrow$ e (clé=-20)
- La plus fréquente lettre en français est "e". La plus fréquente lettre du texte est "i", donc e$\rightarrow$i (clé=4)

Avec une clé de 4 on obtient :

Que j'aime à faire connaitre ce nombre utile aux sages

Cassage du codage César

Soit le message codé :

Uyi n'emqi e jemvi gsrreixvi gi rsqfvi yxmpi eyb wekiw

Deux idées pour décoder :

- Il y a très peu de mots d'une seule lettre en français : a, y
  $\Longrightarrow$ a $\rightarrow$ e (clé=4) ou y $\rightarrow$ e (clé=-20)
- La plus fréquente lettre en français est "e". La plus fréquente lettre du texte est "i", donc e$\rightarrow$i (clé=4)

Avec une clé de 4 on obtient :

Que j'aime à faire connaitre ce nombre utile aux sages

Cassage du codage César

Soit le message codé :

Uyi n'emqi e jemvi gsrreixvi gi rsqfvi yxmpi eyb wekiw

Deux idées pour décoder :

- Il y a très peu de mots d'une seule lettre en français : a, y
  $\Longrightarrow$ a $\rightarrow$ e (clé=4) ou y $\rightarrow$ e (clé=-20)
- La plus fréquente lettre en français est "e". La plus fréquente lettre du texte est "i", donc e$\rightarrow$i (clé=4)

Avec une clé de 4 on obtient :

Que j'aime à faire connaitre ce nombre utile aux sages

Clés symétriques et asymétriques

Cryptographie symétrique : une seule clé secrète

- La même clé secrète utilisée pour coder et décoder
  - un nombre < 30 pour le code de César
  - > 128 bits pour AES (*Advanced Encryption Standard*)
- Quasi-impossible de décoder un message codé si on n'a pas cette clé secrète (AES)
- Le https code les données avec une clé symétrique
  - le plus difficile est de générer une clé symétrique secrète !

Cryptographie asymétrique : une clé publique $\oplus$ clé privée

- Un message codé par la clé publique peut être décodé uniquement par la clé privé
- Un message codé par la clé privé peut être décodé uniquement par la clé publique
- Algo utilisé fréquemment : RSA (Rivest-Shamir-Adleman)

Clés symétriques et asymétriques

Cryptographie symétrique : une seule clé secrète

- La même clé secrète utilisée pour coder et décoder
- Quasi-impossible de décoder un message codé si on n'a pas cette clé secrète (AES)
- Le https code les données avec une clé symétrique
  - le plus difficile est de générer une clé symétrique secrète !

Cryptographie asymétrique : une clé publique $\oplus$ clé privée

- Un message codé par la clé publique peut être décodé uniquement par la clé privé
- Un message codé par la clé privé peut être décodé uniquement par la clé publique
- Algo utilisé fréquemment : RSA (Rivest-Shamir-Adleman)

https : les données echangées par *navigateur web* et le *serveur* sont codées par une clé symétrique secrète déterminée au début :

1. le navigateur génère la clé symétrique secrète (aléatoire)
2. le navigateur code la clé symétrique secrète avec la clé publique du serveur et envoie le message codé au serveur
3. Le serveur décode le message (c'est le seul qui peut faire cela) et récupère la clé symétrique secrète

Comment être certain d'avoir la bonne clé publique du serveur

1. le serveur envoie au navigateur un *certificat de sécurité* (avec la clé publique du serveur+identité serveur+ . . .)
2. ce certificat est codé avec la clé privé d'une Autorité de Certification (AC)
3. le certificat est décodé avec la clé publique de l'AC, connue par tout navigateur
   - La clé publique de l'AC peut ouvrir uniquement des messages codés par la clé privé associé (celle de l'AC)

https : les données echangées par *navigateur web* et le *serveur* sont codées par une clé symétrique secrète déterminée au début :

1. le navigateur génère la clé symétrique secrète (aléatoire)
2. le navigateur code la clé symétrique secrète avec la clé publique du serveur et envoie le message codé au serveur
3. Le serveur décode le message (c'est le seul qui peut faire cela) et récupère la clé symétrique secrète

Comment être certain d'avoir la bonne clé publique du serveur

1. le serveur envoie au navigateur un *certificat de sécurité* (avec la clé publique du serveur+identité serveur+ . . .)
2. ce certificat est codé avec la clé privé d'une Autorité de Certification (AC)
3. le certificat est décodé avec la clé publique de l'AC, connue par tout navigateur
   - La clé publique de l'AC peut ouvrir uniquement des messages codés par la clé privé associé (celle de l'AC)

https : les données echangées par *navigateur web* et le *serveur* sont codées par une clé symétrique secrète déterminée au début :

1. le navigateur génère la clé symétrique secrète (aléatoire)
2. le navigateur code la clé symétrique secrète avec la clé publique du serveur et envoie le message codé au serveur
3. Le serveur décode le message (c'est le seul qui peut faire cela) et récupère la clé symétrique secrète

Comment être certain d'avoir la bonne clé publique du serveur

1. le serveur envoie au navigateur un *certificat de sécurité* (avec la clé publique du serveur+identité serveur+ . . .)
2. ce certificat est codé avec la clé privé d'une Autorité de Certification (AC)
3. le certificat est décodé avec la clé publique de l'AC, connue par tout navigateur
   - La clé publique de l'AC peut ouvrir uniquement des messages codés par la clé privé associé (celle de l'AC)

Et avec le protocole *ssh*

Il n'y a plus d'Autorité de Certification

Authentifier la clé publique du serveur

- Lors d'une première connexion, un message de confirmation est envoyé au client
- On demande au client de faire confiance à une clé publique
  - le résumé de la clé est affiché

```
The authenticity of host 'vlad.cnam.fr' can't be established.
RSA key fingerprint is 6f:e1:07:36:42:b2:0b:36:90:67:31:31:c7:5f:3c:f9.
Are you sure you want to continue connecting (yes/no)? no
```

Les pare-feux : Introduction

- Objectif pare-feu/firewall : séparer le réseau interne des dangers (paquets/trames dangereuses) du monde Internet
- Le pare-feu est souvent un filtre de paquets (`iptables`)

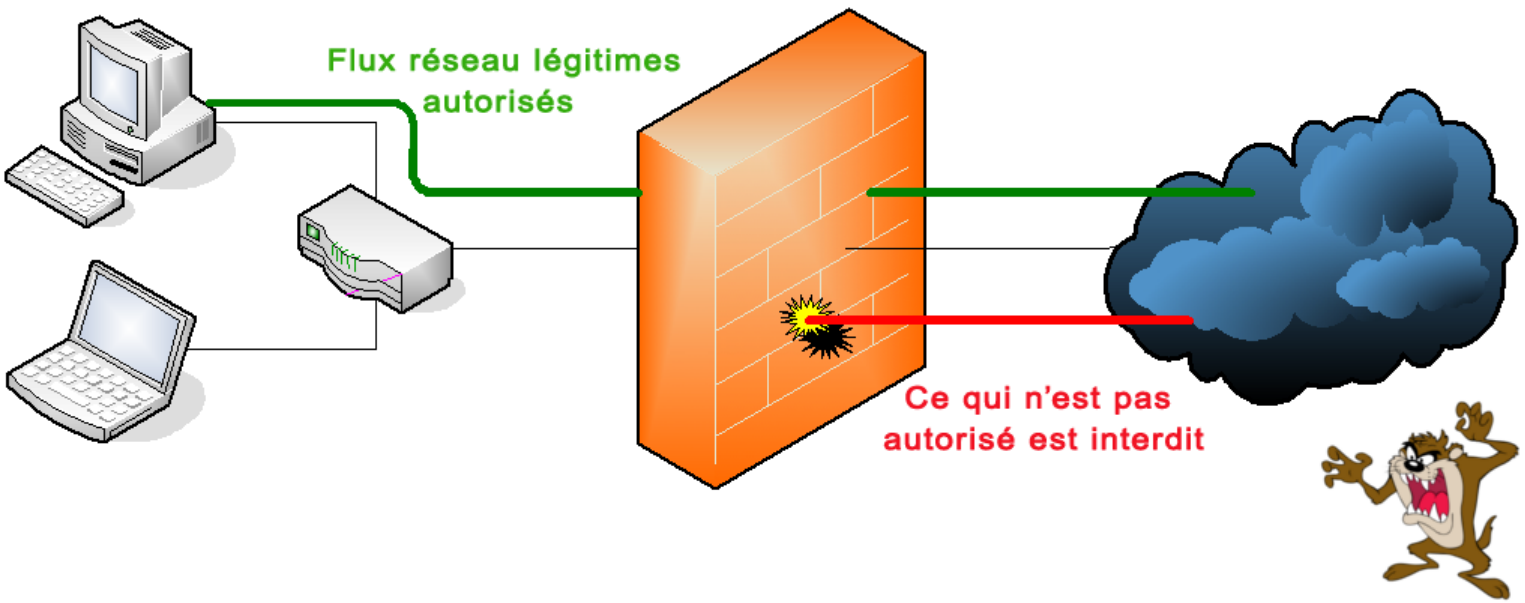
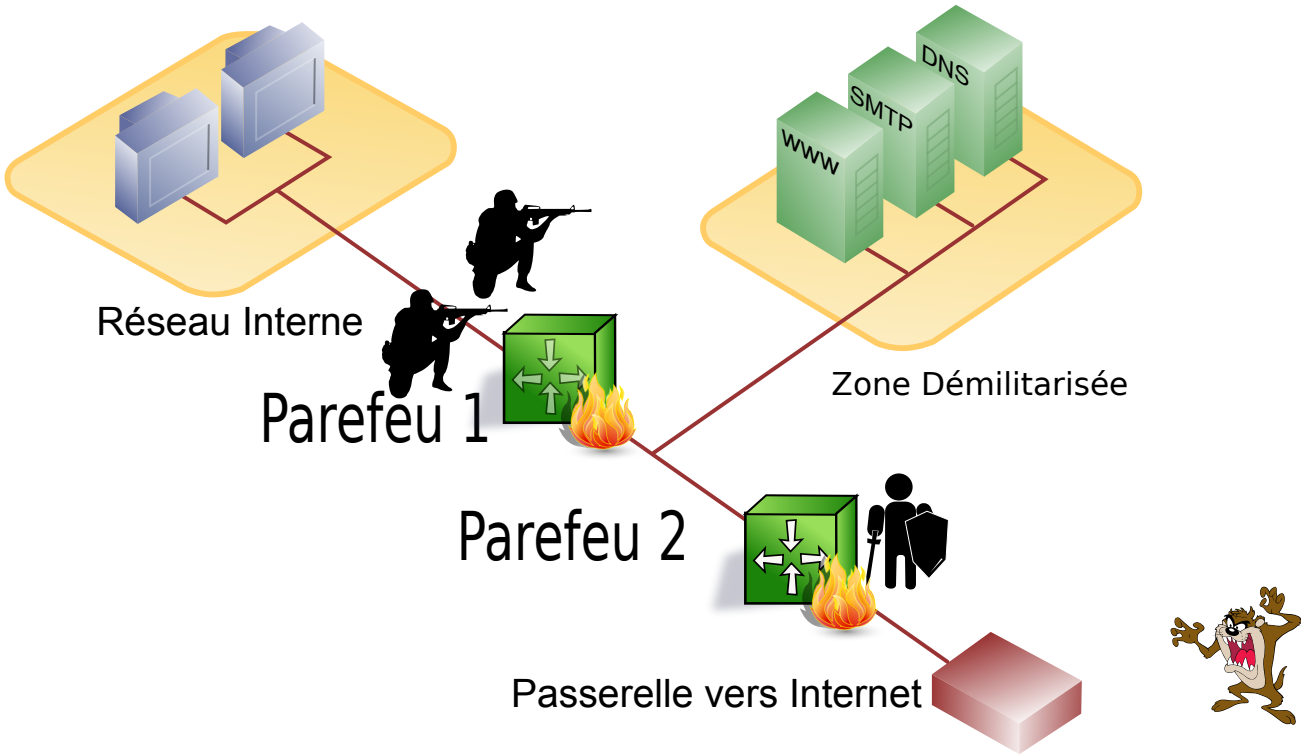


Image due à nanssou.com

Les pares-feu



- Chaque sous-réseau peut avoir son propre pare-feu
- Certains services ont besoin de plus d'accès et ne peuvent pas être trop isolés $\Rightarrow$ ils sont mis dans une zone démilitarisée (DMZ)
  - Au CNAM : possible de se connecter depuis l'extérieur à `www.cnam.fr` (DMZ), mais impossible de se connecter aux machines des salles TP (bien qu'elles aient des IPs publiques)