L'ESSENTIEL

▼ ÉCONOMIE

POLITIQUE. Sarkozy, sans le dire, est entré en campagne pour 2012 en vantant le grand emprunt. **PAGE 4**

CHINE. Les créances douteuses des pouvoirs locaux représentent 1,16 milliard d'euros. **PAGE 5**

OCDE. L'économie mondiale va subir des chocs plus violents et plus fréquents. **PAGE 5**

▼ BUSINESS

NUMÉRIQUE. L'UMP, après le PS, détaille son programme. **PAGE 6**

L'OCDE se penche sur la régulation du Net. **PAGE 6**

EXCLUSIF. THALES perd la confiance de ses clients français. **PAGE 9**

AIR FRANCE cherche un allié en Turquie pour contrer Turkish Airlines. **PAGE 10**

PAYS ÉMERGENTS. SUNART, coentreprise d'Auchan et de Ruentex, veut lever 1 milliard de dollars. **PAGE 11**

GREEN BUSINESS. LA SNCF se lance dans le transport écologique de marchandises. **PAGE 16**

▼ FINANCE

DEXIA cède sa filiale d'assurance-vie turque à MetLife. **PAGE 25**

ENTRETIEN. ÉTIENNE CANIARD, président de la Mutualité française, fait le point sur les enjeux de la contractualisation. **PAGE 26**

CRISE GRECQUE. Le rôle des créanciers privés se précise. **PAGE 26**

Le jackpot du cybercrime organisé

Les pirates informatiques font perdre près de 2,2 millions d'euros par an en moyenne aux entreprises touchées. Une aubaine pour les sociétés de sécurité informatique.

Les revenus de la cybercriminalité représentent des centaines de milliards de dollars par an, dépassant ceux du trafic de drogue.

PAGES 2-3 ET ÉDITORIAL PAGE 38

1. Introduction
   - L'enjeu professionel
   - Exemples d'attaques et de vulnérabilités
2. Pile TCP/IP : rappels et exemples d'attaques
3. Solutions pratique de protection

Pirater les mails des eurodéputés : un “jeu d’enfant” selon un pirate cité par médiapart (2013)

“Avec un ordinateur portable bas de gamme équipé du wifi et quelques connaissances que tout le monde est capable de trouver sur internet, n’importe qui est capable de faire la même chose”

La méthode est simple

1. le pirate s’est installé dans un lieu public à proximité du parlement.
2. il s’est arrangé pour que les téléphones mobiles des gens se trouvant à portée passent par le wifi de son ordinateur pour se connecter à internet
3. Cela lui permet de récupérer les identifiants et mots de passe de ses cibles
   - le téléphone affiche un message abscons, mais beaucoup d’utilisateurs cliquent OK sans le lire

Approche Superficielle

Un logiciel d'attaque permet à un non-spécialiste de "pirater" :

- un moteur de recherche trouve facilement des exemples de logiciels d'attaque (`aircrack`, `hackattack`, etc)

mais :

- un tel logiciel est souvent utilisé sans *trop comprendre le fonctionnement*
- les cibles sont des réseaux pas sécurisés (négligence, pas de données importantes)

Approche Superficielle

Un logiciel d'attaque permet à un non-spécialiste de "pirater" :

- un moteur de recherche trouve facilement des exemples de logiciels d'attaque (`aircrack`, `hackattack`, etc)

mais :

- un tel logiciel est souvent utilisé sans *trop comprendre le fonctionnement*
- les cibles sont des réseaux pas sécurisés (négligence, pas de données importantes)

Approches professionnelle de la sécurité

La *sécurité professionnelle* $\Rightarrow$ l'intention des attaques n'est plus ludique

---

- Les attaquent ciblent des réseaux *très sécurisés* contenant des données sensibles
- Une compréhension générique des réseaux (de couche application) ne suffit plus.

$$\Rightarrow$$

L'objectif de ce cours : étudier la logique et le fonctionnement des protocoles pour comprendre les vulnérabilités de sécurité

1. Introduction
   - L'enjeu professionel
   - Exemples d'attaques et de vulnérabilités

Exemple 1 : Phishing

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

- Envoyer à la victime une page web/mail identique à celui d'un organisme de confiance (la banque de la victime) et lui demander des mots de passe
- Cibles populaires : banques, amazon, paypal, ebay
- Leçons :
  - L'adresse e-mail d'un expéditeur ne garantit pas son identité
    - tout le monde peut envoyer un email avec expéditeur `bill.gates@microsoft.com`
  - attention à vérifier l'adresse web dans la barre d'adresse du navigateur web

Exemple 1 : Phishing

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

- Envoyer à la victime une page web/mail identique à celui d'un organisme de confiance (la banque de la victime) et lui demander des mots de passe
- Cibles populaires : banques, amazon, paypal, ebay
- Leçons :
  - L'adresse e-mail d'un expéditeur ne garantit pas son identité
    - tout le monde peut envoyer un email avec expéditeur `bill.gates@microsoft.com`
  - attention à vérifier l'adresse web dans la barre d'adresse du navigateur web

Exemple 2 : Failles dans les applications Web

Facebook offre des prix pour ceux qui trouvent des failles de sécurités dans les applications *third-party* (écrites par des utilisateurs tiers)

- record détenu par un homme qui a touché 7000$ pour six failles
- Facebook offre 500$ au minimum mais certaines failles valent plus
- Les vulnérabilités importantes (*zero day*) : une valeur beaucoup plus élevée *sur le marché noir*
  - des personnes souhaitant les exploiter peuvent payer des sommes bien plus importantes.
- Google a distribué 100.000$ en 2011 pour des vulnérabilités dans le produit Chrome

Exemple 2 : Failles dans les applications Web

Facebook offre des prix pour ceux qui trouvent des failles de sécurités dans les applications *third-party* (écrites par des utilisateurs tiers)

- record détenu par un homme qui a touché 7000$ pour six failles
- Facebook offre 500$ au minimum mais certaines failles valent plus
- Les vulnérabilités importantes (*zero day*) : une valeur beaucoup plus élevée *sur le marché noir*
  - des personnes souhaitant les exploiter peuvent payer des sommes bien plus importantes.
- Google a distribué 100.000$ en 2011 pour des vulnérabilités dans le produit Chrome

Exemple 3 : Le Déni de Service 1

Cette attaque est souvent distribuée. Un exemple :

- Si la machine A exécute "`ping` B", B répond avec un *ping reply*
- Si les machines $A_1, A_2, \ldots A_n$ sont infectées est programmées à exécuter "ping B" m fois

  $\Rightarrow$ B doit envoyer $n \cdot m$ réponses

  $\Rightarrow$ Si $n > 1.000.000$ et $m = 100$, B doit envoyer des centaines de millions de *ping reply* $\Longrightarrow$ B ne peut plus gérer d'autres demandes légitimes et subit un déni de service.

Exemple 3 : Le Déni de Service 1

Cette attaque est souvent distribuée. Un exemple :

- Si la machine A exécute "`ping` B", B répond avec un *ping reply*
- Si les machines $A_1, A_2, \ldots A_n$ sont infectées est programmées à exécuter "ping B" m fois

  $\Rightarrow$ B doit envoyer $n \cdot m$ réponses

  $\Rightarrow$ Si $n >$ 1.000.000 et $m = 100$, B doit envoyer des centaines de millions de *ping reply* $\Longrightarrow$ B ne peut plus gérer d'autres demandes légitimes et subit un déni de service.

Exemple 3 : Le Déni de Service 2

Certaines attaques Déni de Service sont renommées

Attaque historique 1 7-8 février 2000, *Yahoo, Amazon, Ebay et Cnn.com* ont été quasi-inaccessibles $\Rightarrow$ pertes de millions

Attaque historique 2 8 décembre 2010, *MasterCard et Visa* quasi-inacces à cause du groupe “Anonymous” (solidarité avec Wiki-Leaks)

Il y a des pirates spécialisés dans la “levée” de *zombies* (les machines $A_1, A_2, \ldots A_n$ infectées pour installer des agents d'attaque)

- Cette “armée de zombies” peut être ensuite louée à d'autres pirates

Exemple 3 : Le Déni de Service 2

Certaines attaques Déni de Service sont renommées

Attaque historique 1 7-8 février 2000, *Yahoo, Amazon, Ebay et Cnn.com* ont été quasi-inaccessibles $\Rightarrow$ pertes de millions

Attaque historique 2 8 décembre 2010, *MasterCard et Visa* quasi-inacces à cause du groupe "Anonymous" (solidarité avec Wiki-Leaks)

Il y a des pirates spécialisés dans la "levée" de *zombies* (les machines $A_1, A_2, \ldots A_n$ infectées pour installer des agents d'attaque)

- Cette "armée de zombies" peut être ensuite louée à d'autres pirates

Exemple 4 : Failles de programmation

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

```
1 int main (int argc, char **argv)
2 {
3     char buf [1000] ;
4     strcpy (buf,argv [1]) ;
5 }
```

Pour mémoire :*Linux, Windows et Mac OS* sont écrits en *C/C++*

Exécution :

```
@$ ./demo aaa...1010 fois...aaaaaaaaaaaaaa
Segmentation fault
```

→ Erreur de *buffer overflow* : dépassement de tampon/mémoire

→ Ce type de problème apparaît très souvent lors de la réception de paquets, trames et autres données de réseaux

Exemple 4 : Failles de programmation

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

```
1 int main (int argc, char **argv)
2 {
3     char buf [1000] ;
4     strcpy (buf,argv [1]) ;
5 }
```

Pour mémoire :*Linux, Windows et Mac OS* sont écrits en *C/C++*

Exécution :

```
@$ ./demo aaa...1010 fois...aaaaaaaaaaaaaa
Segmentation fault
```

→ Erreur de *buffer overflow* : dépassement de tampon/mémoire
→ Ce type de problème apparaît très souvent lors de la réception de paquets, trames et autres données de réseaux

Exemple 5 : 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 SEs (Windows/Linux)
- Un simple *ping* pouvait provoquer un crash d'une machine cible (Unix, Linux, MacOS, Windows)

Exemple 6 : La faille CON-CON (*Windows*)

- *Windows* : les noms de périphériques (con, lpt1) sont réservés
  - Impossible de créer un dossier "aux", "con" ou "lpt1"
  - Lorsque *Windows* détecte ces noms de dossiers, il cherche le périphérique concerné
- L'utilisation d'un tel nom deux fois dans un chemin d'accès provoque un crash de certains versions de *Windows*

$\Rightarrow$ Un accès depuis le réseaux *Windows* à *ordinateurCible*\dossierPartagé\con\con\

plante *ordinateurCible*

Exemple 8 : Falsification de certificats

Le 29 août 2011, Google a annoncé une tentative d'espionnage généralisée visant principalement les internautes iraniens. Principes :

- Toute connexion `https` à `mail.google.com` est codé avec *la clé publique Google*
  - Uniquement Google peut lire un message codé avec sa clé privée
- Avec un faux certificat, le navigateur utilise une fausse clé publique
- Le trafic `https` vers google et intercepté par "l'homme du milieu" qui offre une page de login comme celle de Google
- L'utilisateur tape son mots de passe qui est codé par la fausse clé et décodé par l'homme du milieu

Exemple 9 : Spoofing, Sniffing et Pharming

- IP Spoofing : une machine *A* envoie des paquets IP en utilisant comme source l'adresse IP de *B*.
  - *A* peut "emprunter" l'identité de *B*
  - serveur DHCP : *A* utilise l'adresse MAC du *B*, $\Rightarrow$ le protocole DHCP affecte l'adresse IP de *B* à *A*
- Sniffing : une machine *C* intercepte le trafic entre *A* et *B*
- DNS Pharming : la connexion du client vers un site est détournée vers un faux site
  - Le faux site est en apparence identique au vrais site

Exemple 9 : Spoofing, Sniffing et Pharming

- IP Spoofing : une machine A envoie des paquets IP en utilisant comme source l'adresse IP de B.
  - A peut "emprunter" l'identité de B
  - serveur DHCP : A utilise l'adresse MAC du B, $\Rightarrow$ le protocole DHCP affecte l'adresse IP de B à A
- Sniffing : une machine C intercepte le trafic entre A et B
- DNS Pharming : la connexion du client vers un site est détournée vers un faux site
  - Le faux site est en apparence identique au vrais site

Exemple 9 : Spoofing, Sniffing et Pharming

- IP Spoofing : une machine A envoie des paquets IP en utilisant comme source l'adresse IP de B.
  - A peut "emprunter" l'identité de B
  - serveur DHCP : A utilise l'adresse MAC du B, $\Rightarrow$ le protocole DHCP affecte l'adresse IP de B à A
- Sniffing : une machine C intercepte le trafic entre A et B
- DNS Pharming : la connexion du client vers un site est détournée vers un faux site
  - Le faux site est en apparence identique au vrais site

Exemple 11 : Kevin Mitnick – une histoire célèbre

- En 1983, Kevin Mitnick s'est connecté au réseau ARPAnet (ancêtre de l'Internet) de l'université de Californie et est a obtenu un accès illégal à tous les fichiers du Pentagone
- Plus tard, il est le premier à avoir mis en pratique le "IP Spoofing" en exploitant une faille du TCP de l'époque
  - Exemple : l'attaque "IP Spoofing" ci-dessus a été mené le 25 décembre.

Exemple 11 : Kevin Mitnick – une histoire célèbre

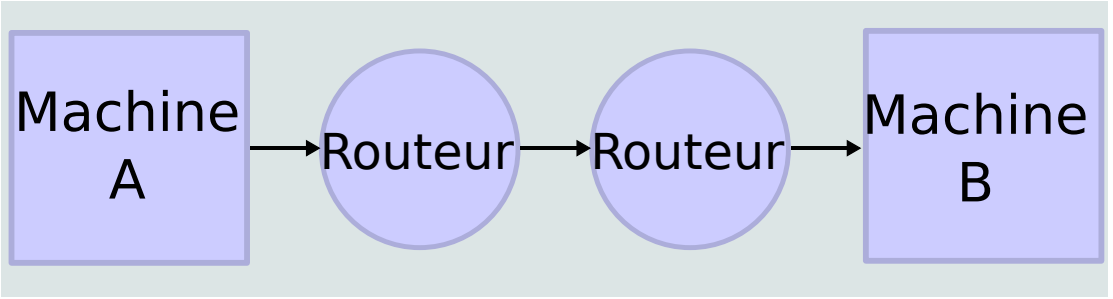
- En plus de Pentagone, il est célèbre pour avoir accédé aux bases de données de Bell, Fujitsu, Motorola, Nokia et Sun Microsystems.
- Il est le premier pirate à avoir figuré dans la liste des dix criminels les plus recherchés par le FBI aux États-Unis.
- Il est aujourd'hui *consultant* en sécurité informatique

Rappels rapides : pile TCP/IP

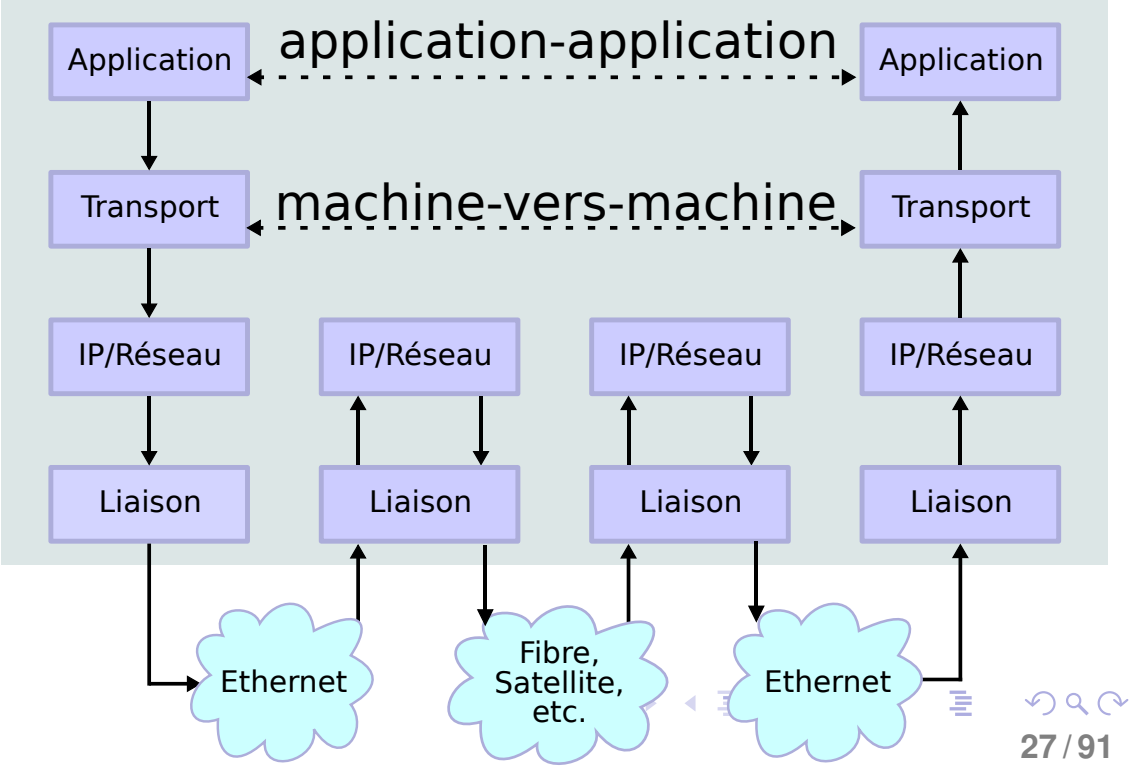
Quatre couches de protocoles qui s'appuient sur la couche physique :

1. Couche Application (Firefox, MSN, FTP)
2. Couche Transport (TCP/UDP)
3. Couche IP/Réseaux
4. Couche Liaison

Topologie Réseau



Flux de données



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

Pour se connecter à un site :

- Le navigateur interroge son serveur DNS (Domain Name Server)
  - Voir `/etc/resolv.conf` sous Linux
  - Voir `/etc/hosts` sous Linux
- 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 grave : ce processus n'est pas trop sécurisé ⇒ si la réponse DNS est fausse, la navigation Web est compromise

L'attaque Pharming : faire le serveur DNS envoyer une fausse réponse

Approches possibles d'attaque

1. Modifier l'application du serveur local DNS : exploiter des failles de programmation de celle-ci
2. Injecter des informations erronées dans le serveur DNS ⇒ une attaque très puissante : DNS Cache Poisoning,
3. Modifier les configurations de routage pour diriger le trafic DNS par une machine pirate
   - Les messages DNS ne sont pas codés ⇒ attaque `Man In The Middle` possible, i.e. un "homme du milieu" peut modifier l'information DNS

L'attaque Pharming : faire le serveur DNS envoyer une fausse réponse

Approches possibles d'attaque

1. Modifier l'application du serveur local DNS : exploiter des failles de programmation de celle-ci
2. Injecter des informations erronées dans le serveur DNS $\Rightarrow$ une attaque très puissante : **DNS Cache Poisoning**,
3. Modifier les configurations de routage pour diriger le trafic DNS par une machine pirate
   - Les messages DNS ne sont pas codés $\Rightarrow$ attaque `Man In The Middle` possible, i.e. un "homme du milieu" peut modifier l'information DNS

L'attaque Pharming : faire le serveur DNS envoyer une fausse réponse

Approches possibles d'attaque

1. Modifier l'application du serveur local DNS : exploiter des failles de programmation de celle-ci
2. Injecter des informations erronées dans le serveur DNS $\Rightarrow$ une attaque très puissante : **DNS Cache Poisoning**,
3. Modifier les configurations de routage pour diriger le trafic DNS par une machine pirate
   - Les messages DNS ne sont pas codés $\Rightarrow$ attaque `Man In The Middle` possible, i.e. un "homme du milieu" peut modifier l'information DNS

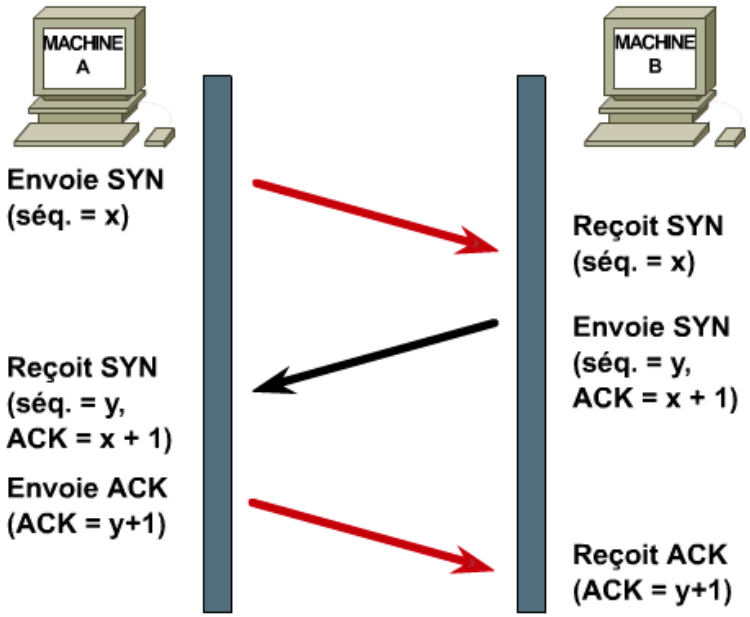
Une connexion TCP comporte trois étapes :

1. Établissement de la connexion TCP.
2. Transfert de données
3. Fermeture de la connexion TCP.

Ce canal de communication est une socket défini par deux adresses IP:PORT, par exemple :

`123.123.123.123:20312` ⇔ `207.142.131.203:80`

TCP phase 1 : initialisation connexion

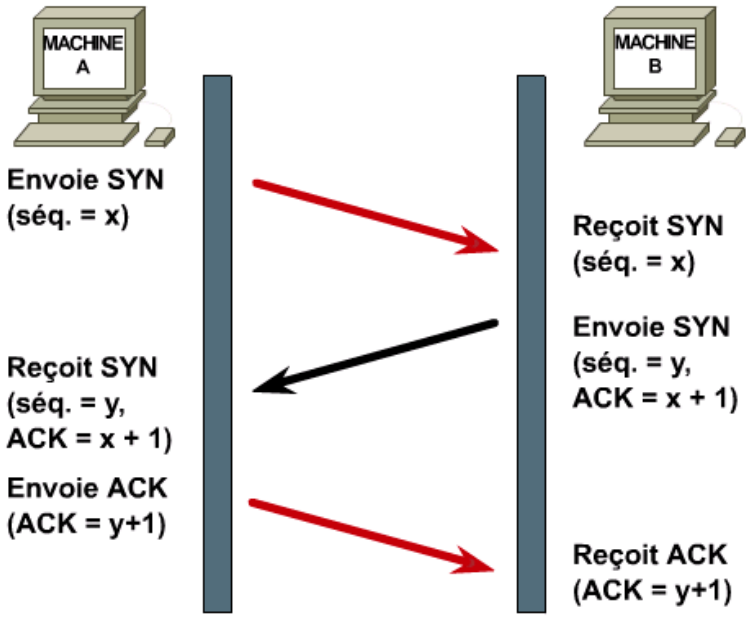


Attaque SYN Flood : un pirate supprime la dernière étape. [2]

- pour le serveur, la connexion reste semi-ouverte et consomme des ressources (mémoire, temps processeur, etc.).
- beaucoup de connexions semi-ouvertes ⇒ serveur surchargé
  - en 1996, un FAI américain a été paralysé par un Syn flood

2. `fr.wikipedia.org/wiki/SYN_flood`

TCP phase 1 : initialisation connexion



Attaque SYN Flood : un pirate supprime la dernière étape. [2]

- pour le serveur, la connexion reste semi-ouverte et consomme des ressources (mémoire, temps processeur, etc.).
- beaucoup de connexions semi-ouvertes $\Rightarrow$ serveur surchargé
  - en 1996, un FAI américain a été paralysé par un Syn flood

2. `fr.wikipedia.org/wiki/SYN_flood`

TCP : phase 3 : libération de connexion

- Protocole simpliste en deux temps :

  $A \rightarrow B$ J'ai terminé. Es-tu d'accord ?

  $B \rightarrow A$ D'accord, j'ai fini ; au revoir.

- Problème des deux armées : comment les armées rouges vont pouvoir attaquer exactement au même moment ?
  - Armée $A \rightarrow$ armée B : "Nous allons attaquer à 17h"
  - Réponse $B \rightarrow$ armée A : "Je confirme, attaquer à 17h"
  - C'est sûr que A reçoit la confirmation ?

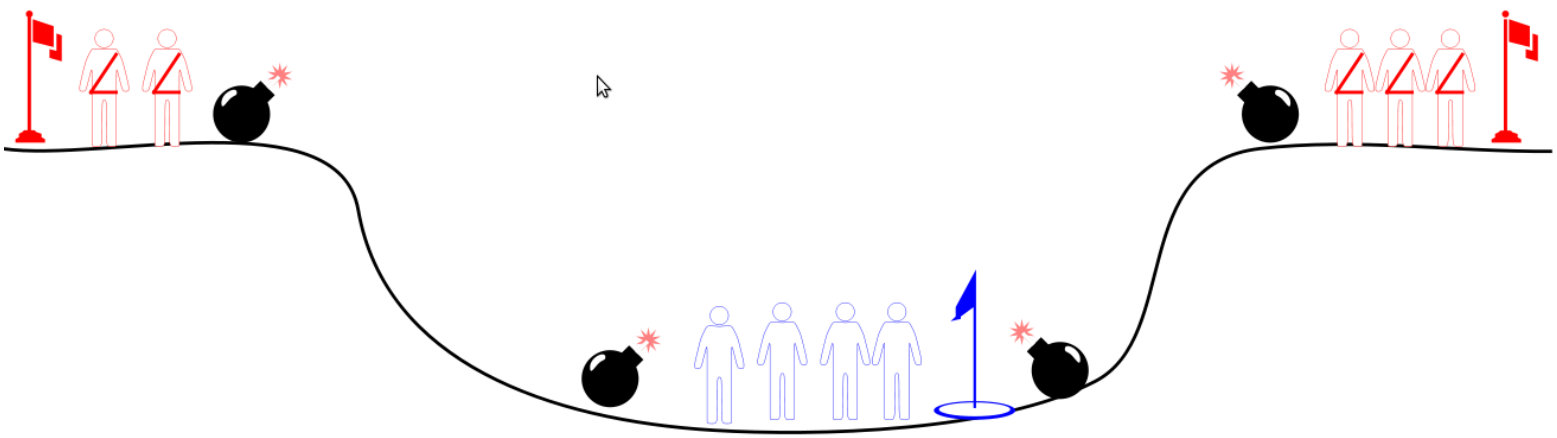
TCP : phase 3 : libération de connexion

- Protocole simpliste en deux temps :

  $A \rightarrow B$ J'ai terminé. Es-tu d'accord ?

  $B \rightarrow A$ D'accord, j'ai fini ; au revoir.

- Problème des deux armées : comment les armées rouges vont pouvoir attaquer exactement au même moment ?
  - Armée $A \rightarrow$ armée B : "Nous allons attaquer à 17h"
  - Réponse $B \rightarrow$ armée A : "Je confirme, attaquer à 17h"
  - C'est sûr que A reçoit la confirmation ?



Vulnérabilités Couche Réseau

Un paquet IP peut transiter des dizaines de routeurs pour arriver à destination

- Chaque routeur peut “cacher” un pirate qui sniffe la communication
- Données pas codées par les couches supérieurs ⇒ tout routeur peut forger de faux paquets IP (l’attaque du l’homme du milieu)

Certaines protocoles de routage peuvent avoir des failles

- RIP (Routing Information Protocol) : ce protocole permet de changer ses routes quand un serveur “propose” de “meilleures” routes
- Reroutage BGP (Border Gateway Protocol) : modifier les routes des autres routeurs pour faire le trafic passer par une machine pirate
  - Exemple réel : le traffic vers `youtube` bloqué 2h en 2008 [3]

3. `www.ripe.net/internet-coordination/news/industry-developments/youtube-hijacking-a-ripe-ncc-ris-case-study`

Vulnérabilités Couche Réseau

Un paquet IP peut transiter des dizaines de routeurs pour arriver à destination

- Chaque routeur peut “cacher” un pirate qui sniffe la communication
- Données pas codées par les couches supérieurs $\Rightarrow$ tout routeur peut forger de faux paquets IP (l’attaque du l’homme du milieu)

Certaines protocoles de routage peuvent avoir des failles

- RIP (Routing Information Protocol) : ce protocole permet de changer ses routes quand un serveur “propose” de “meilleures” routes
- Reroutage BGP (Border Gateway Protocol) : modifier les routes des autres routeurs pour faire le trafic passer par une machine pirate
  - Exemple réel : le traffic vers `youtube` bloqué 2h en 2008 [3]

3. `www.ripe.net/internet-coordination/news/industry-developments/youtube-hijacking-a-ripe-ncc-ris-case-study`

Vulnérabilités Couche Réseau

Un paquet IP peut transiter des dizaines de routeurs pour arriver à destination

- Chaque routeur peut “cacher” un pirate qui sniffe la communication
- Données pas codées par les couches supérieurs $\Rightarrow$ tout routeur peut forger de faux paquets IP (l'attaque du l'homme du milieu)

Certaines protocoles de routage peuvent avoir des failles

- RIP (Routing Information Protocol) : ce protocole permet de changer ses routes quand un serveur “propose” de “meilleures” routes
- Reroutage BGP (Border Gateway Protocol) : modifier les routes des autres routeurs pour faire le trafic passer par une machine pirate
  - Exemple réel : le traffic vers `youtube` bloqué 2h en 2008 [3]

3. `www.ripe.net/internet-coordination/news/industry-developments/youtube-hijacking-a-ripe-ncc-ris-case-study`

Vulnérabilités Couche Réseau

Un paquet IP peut transiter des dizaines de routeurs pour arriver à destination

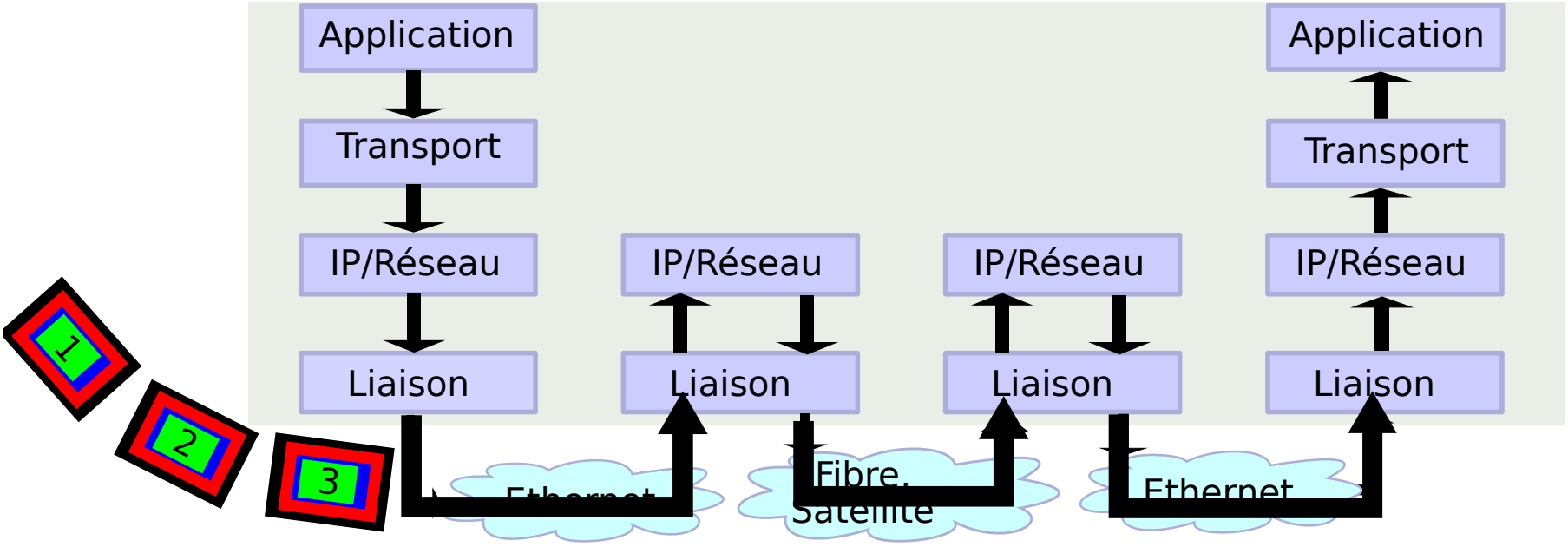
- Chaque routeur peut “cacher” un pirate qui sniffe la communication
- Données pas codées par les couches supérieurs $\Rightarrow$ tout routeur peut forger de faux paquets IP (l'attaque du l'homme du milieu)

Certaines protocoles de routage peuvent avoir des failles

- RIP (Routing Information Protocol) : ce protocole permet de changer ses routes quand un serveur “propose” de “meilleures” routes
- Reroutage BGP (Border Gateway Protocol) : modifier les routes des autres routeurs pour faire le trafic passer par une machine pirate
  - Exemple réel : le traffic vers `youtube` bloqué 2h en 2008 [3]

3. `www.ripe.net/internet-coordination/news/industry-developments/youtube-hijacking-a-ripe-ncc-ris-case-study`

Rappels : 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
- Le protocole ARP fait la connexion IP $\rightarrow$ MAC.
  - Etant donnée une adresse IP de notre sous-réseau, quelle est son adresse MAC ?
  - Un point très sensible de la sécurité des réseaux locaux

Vulnérabilités Couche Liaison 1

- Couche liaison : envoi de trames entre deux machines *A* et *B* en connexion directe (sans routeur intermédiaire)
- Toute carte réseau possède une adresse MAC (Media Access Control) : 6 octets, ex hexa : `00:12:3F:DD:A2:17`
  - fournie par le producteur mais il est possible de la changer

Correspondances entre adresses IP et adresses MAC

- $IP \rightarrow MAC$: protocole ARP (Address Resolution Protocol)
- $MAC \rightarrow IP$: protocole RARP (Reverse ARP)

Vulnérabilités Couche Liaison 1

- Couche liaison : envoi de trames entre deux machines *A* et *B* en connexion directe (sans routeur intermédiaire)
- Toute carte réseau possède une adresse MAC (Media Access Control) : 6 octets, ex hexa : `00:12:3F:DD:A2:17`
  - fournie par le producteur mais il est possible de la changer

Correspondances entre adresses IP et adresses MAC

- $IP \rightarrow MAC$: protocole ARP (Address Resolution Protocol)
- $MAC \rightarrow IP$: protocole RARP (Reverse ARP)

Vulnérabilités couche Liaison 2

- Dans un réseau Wifi/Ethernet, plusieurs machines C, D, E ... sont en connexion directe avec l'émetteur A
- Une trame envoyée vers MAC_B devrait être reçue uniquement par B mais pas de contrainte physiques pour assurer cela
- Pour faire capter les trames destinées à d'autres machines, une carte peut passer en :
  - Mode Moniteur en Wifi (`iwconfig`)
  - Mode "Promiscuité" en Ethernet (`ifconfig`)

- Les signaux électriques sur un câble Ethernet peuvent être sniffés
  - Solution de protection : pressuriser les câbles

Vulnérabilités couche Liaison 2

- Dans un réseau Wifi/Ethernet, plusieurs machines C, D, E ... sont en connexion directe avec l'émetteur A
- Une trame envoyée vers MAC_B devrait être reçue uniquement par B mais pas de contrainte physiques pour assurer cela
- Pour faire capter les trames destinées à d'autres machines, une carte peut passer en :
  - Mode Moniteur en Wifi (`iwconfig`)
  - Mode "Promiscuité" en Ethernet (`ifconfig`)

- Les signaux électriques sur un câble Ethernet peuvent être sniffés
  - Solution de protection : pressuriser les câbles

Vulnérabilité couche Liaison 3

Le protocole ARP n'est pas sécurisé, mais plutôt basé sur la confiance

- Pour trouver MAC_B, la machine A envoie en diffusion une demande ARP :

  `Je veux la MAC d'` IP_B`. Répondez à` MAC_A

- Toutes les machines $C, D, E, F, G \ldots$ en connexion directe captent cette trame mais uniquement B doit répondre
- Scénario ARP Poisoning en deux étapes :
  1. une autre machine C envoie une réponse :

     IP_B `se trouve à` MAC_C
  2. C devient une passerelle entre A et B : tous les paquets envoyés de A à B arrivent d'abord à C et sont ensuite transférés vers B

Vulnérabilité couche Liaison 3

Le protocole ARP n'est pas sécurisé, mais plutôt basé sur la confiance

- Pour trouver MAC_B, la machine A envoie en diffusion une demande ARP :

  `Je veux la MAC d'` IP_B `. Répondez à` MAC_A

- Toutes les machines $C, D, E, F, G \ldots$ en connexion directe captent cette trame mais uniquement B doit répondre
- Scénario ARP Poisoning en deux étapes :

  1. une autre machine C envoie une réponse :

     IP_B `se trouve à` MAC_C

  2. C devient une passerelle entre A et B : tous les paquets envoyés de A à B arrivent d'abord à C et sont ensuite transférés vers B

1. Introduction

2. Pile TCP/IP : rappels et exemples d'attaques

3. Solutions pratique de protection
   - Éléments de Cryptographie
   - Routage et filtrage via `Iptables`

L'importance de la 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
  - *Exemple* : les attaques sniffing devient beaucoup moins efficaces si les données sniffées sont codées
- La cryptographie est divisée en deux parties :
  1. la cryptographie symétrique : une seule clef secrète
  2. la cryptographie asymétrique : une clé publique $\oplus$ clé privée

D'autres facettes de la cryptologie

- Les techniques de cryptographie assurent la confidentialité
- La cryptographie est juste une partie de la cryptologie, une science à plusieurs facettes :
  - La stéganographie – l'art de la dissimulation : faire passer inaperçu un message dans un autre
    - (Pommes : 4.0**7** ; Tomates : 12.0**5** ; Poires : 5.0**3** ; Persil : 3.2**3**)
    - En informatique, il est possible d'envoyer des messages cachés dans les images (dans les fichiers `.jpg`)
  - L'authentification – l'assurance de l'identité d'une personne
    - Exemple : une signature électronique (il n'y pas forcement de confidentialité dans une signature)
  - La non-répudiation – la preuve/trace des activités passés
    - assurée par les logs et sauvegardes automatiques

D'autres facettes de la cryptologie

- Les techniques de cryptographie assurent la confidentialité
- La cryptographie est juste une partie de la cryptologie, une science à plusieurs facettes :
  - La stéganographie – l'art de la dissimulation : faire passer inaperçu un message dans un autre
    - (Pommes : 4.0**7** ; Tomates : 12.0**5** ; Poires : 5.0**3** ; Persil : 3.2**3**)
    - En informatique, il est possible d'envoyer des messages cachés dans les images (dans les fichiers `.jpg`)
  - L'authentification – l'assurance de l'identité d'une personne
    - Exemple : une signature électronique (il n'y pas forcement de confidentialité dans une signature)
  - La non-répudiation – la preuve/trace des activités passés
    - assurée par les logs et sauvegardes automatiques

Principe :

Message Clair
« RV le 24/12 à 20H »

Chiffrement

Message codé
« gdsg cgdtxx »

Déchiffrement

Message Clair
« RV le 24/12 à 20H »

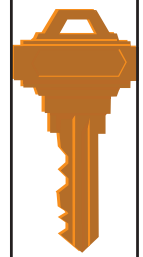
Présentation due à Nathalie Souf

1) brouillage
ex :Chiffrement de Jules César

Joyeux Noel

Décalage de 2 lettres « avant »

Lqagwz Pqgn



Décalage de 2 lettres «arrière»

Joyeux Noel

Cassage du chiffrement

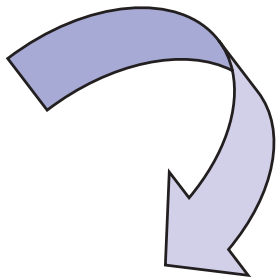
A partir du tableau de fréquence des lettres.

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

1u	2g
3y	1o
9i	3r
1n	2x
6e	2v
3m	1p
1j	1b
3v	2w

Le e est le plus fréquent en français

=> i = e

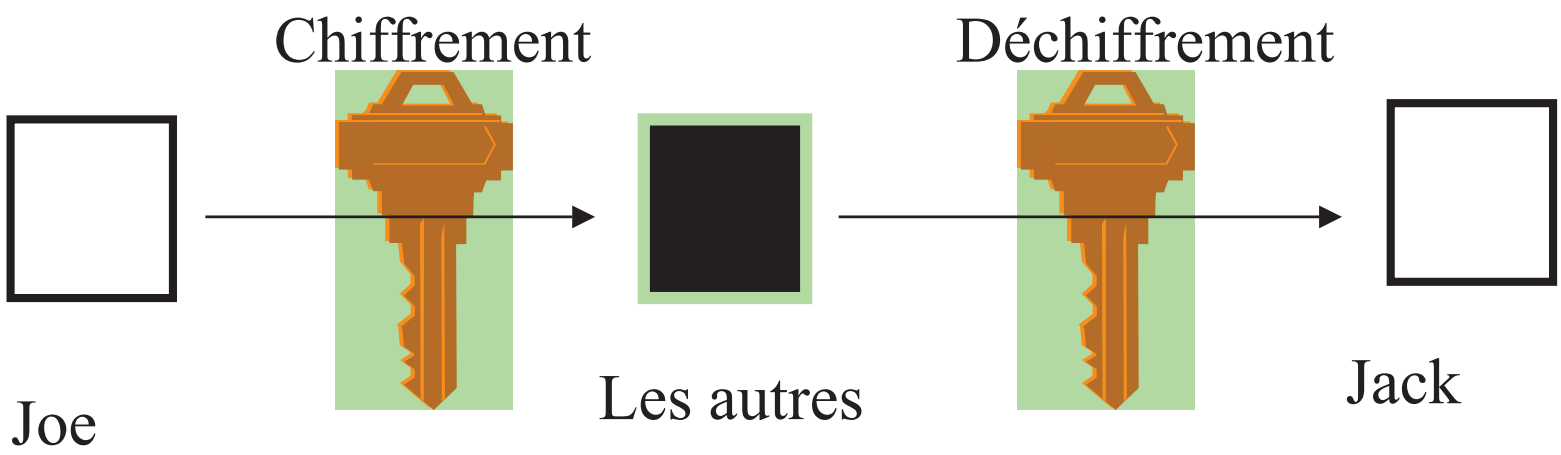


Décalage de 4 en arrière

Que j 'aime à faire connaître ce nombre utile aux sages.

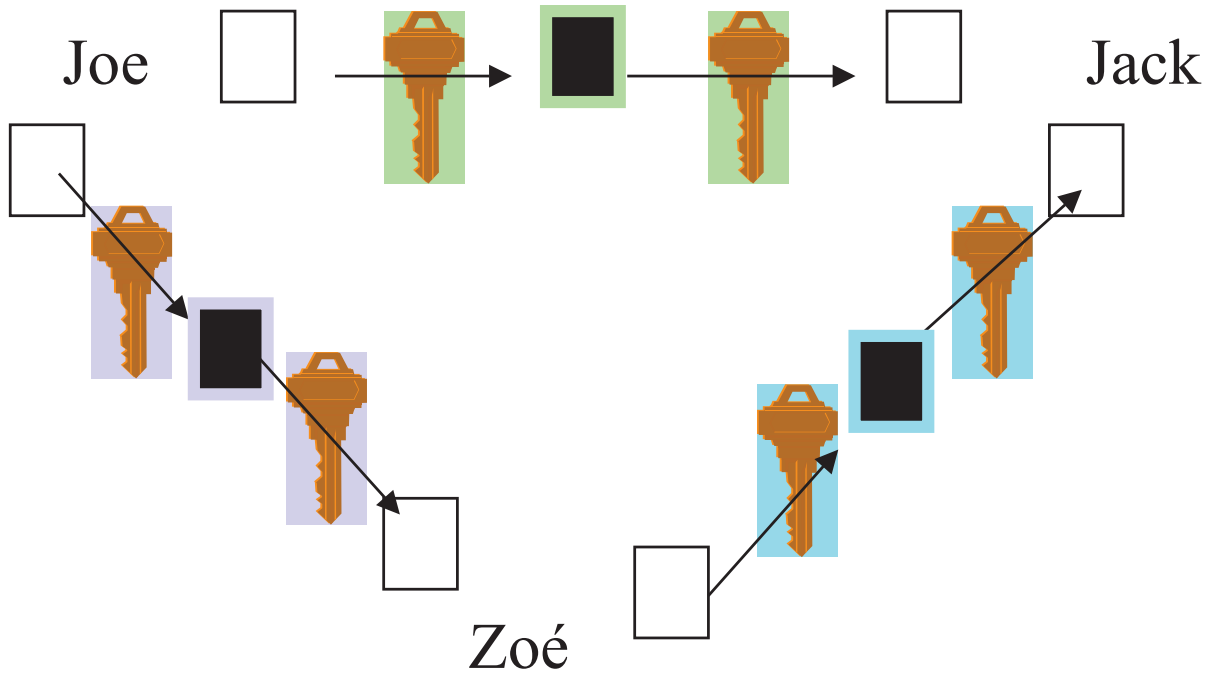
Chiffrage à clé symétrique
Clé privée

- Exemple DES (data Encryption Standard)
- C 'est la même clé (secrète) qui sert à encoder et à décoder



Cryptographie à clé secrète

- Problème 1: le secret de la clé
- Problème 2 : le nombre de clés



Clés asymétriques

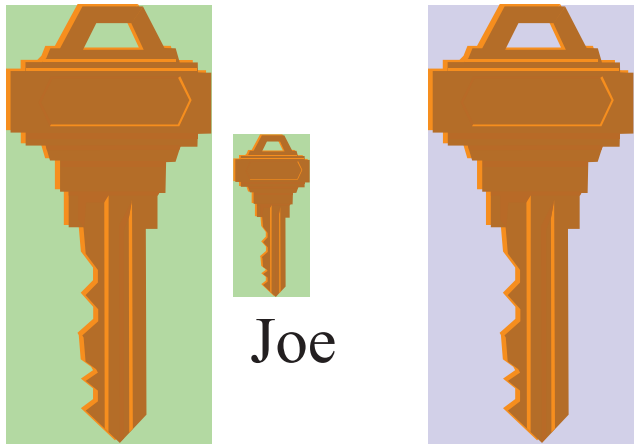
2 clés : CLE1, clé2

- un message encodé avec CLE1 sera décodé avec clé2
- un message décodé avec clé2 aura été codé avec CLE1
- Pas moyen de déduire l ’une de l ’autre

  (propriétés de grands nombres premiers)

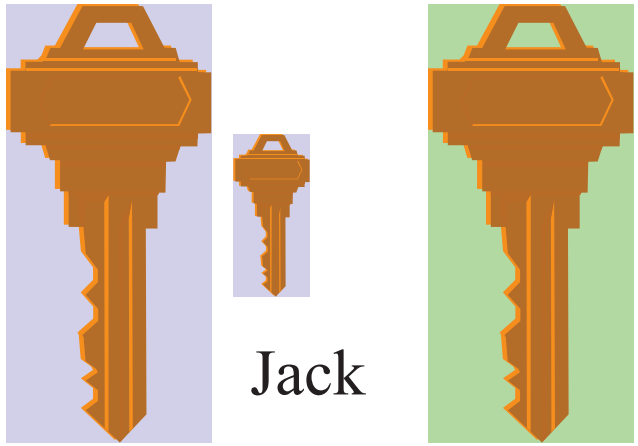
Ex RSA (Rivest Shamir Akermann)

Clés asymétriques
Clés publiques / clés privées



Joe possède une clé publique et une clé privée

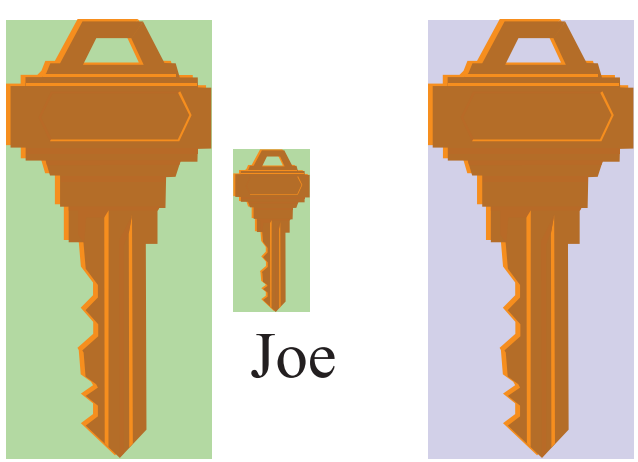
Joe connaît la clé publique de Jack

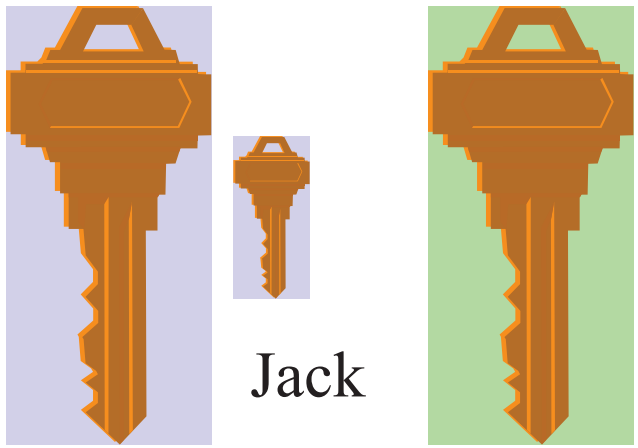


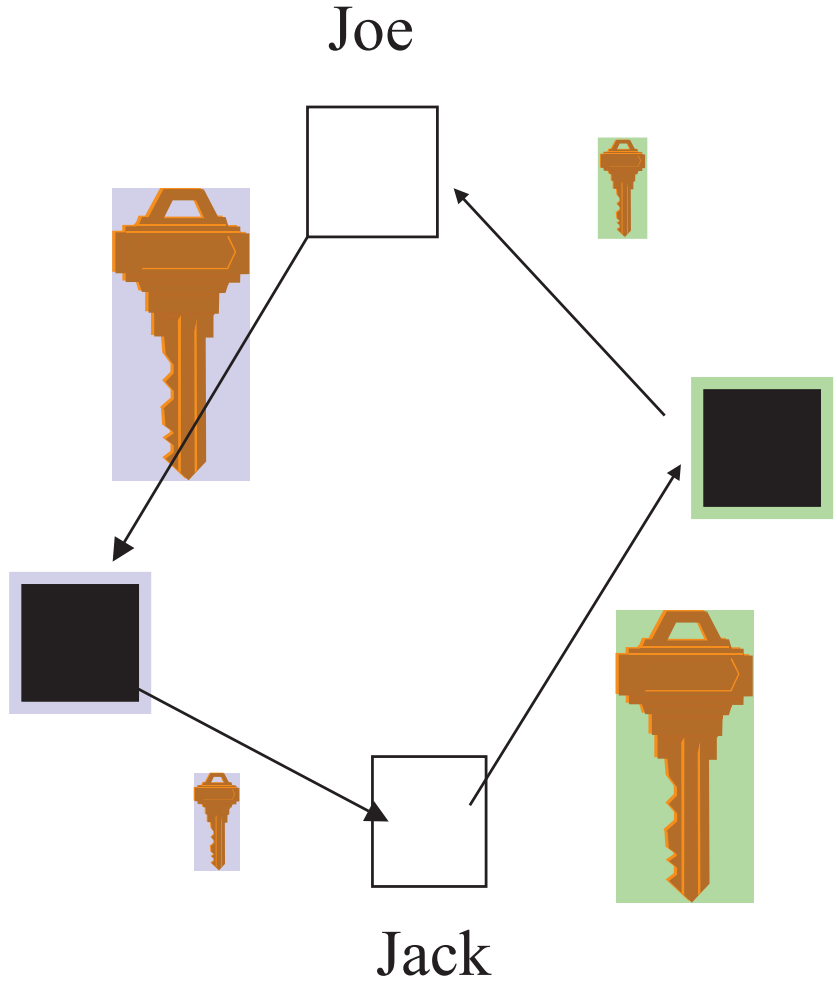
Jack possède une clé publique et une clé privée

Jack connaît la clé publique de Joe

Clés asymétriques
Clés publiques / clés privées







Plusieurs partenaires

Un gestionnaire de clés publiques

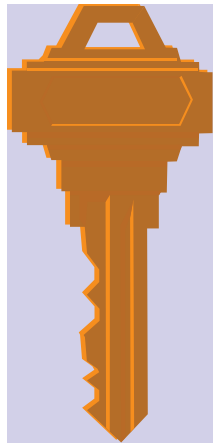
Joe 

Jack 

Zoe 

Chacun garde sa clé privée

Chiffrement



Un gestionnaire de clés publiques

Joe 

Jack 

Joe veut encoder les données qu'il envoie à Jack

1) Joe crypte le message avec la clé publique de Jack
2) Jack le décode avec sa clé privée

Authentification

Un gestionnaire de clés publiques

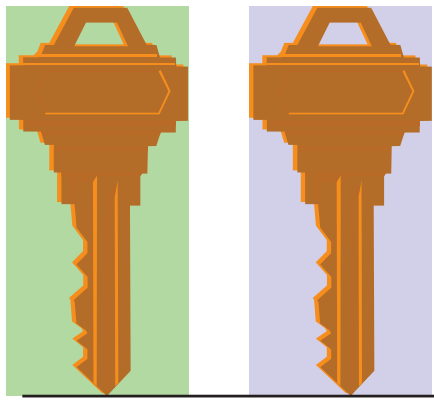
Joe 

Jack 

Joe veut parler (en clair) à Jack mais en étant sûr que c'est bien Jack

1) Joe envoie à Jack un message aléatoire 
2) Jack le renvoie en l'ayant codé avec sa clé privée
3) Joe le décode et vérifie que c'est bien le message d'origine

Intégrité



Un gestionnaire de clés publiques

Joe 

Jack 

Joe veut parler (en clair) à Jack mais en étant sûr que personne ne modifiera ses données

1) Joe envoie à Jack son message +(un résumé calculé sur le message et codé )

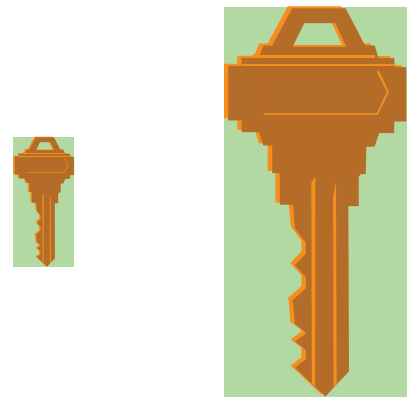
2) Jack fait le même calcul de résumé sur le message, décode le résumé de Joe et vérifie que les 2 coïncident

Et sur le Web ??

Exemple SSL (Secure Socket Layer)

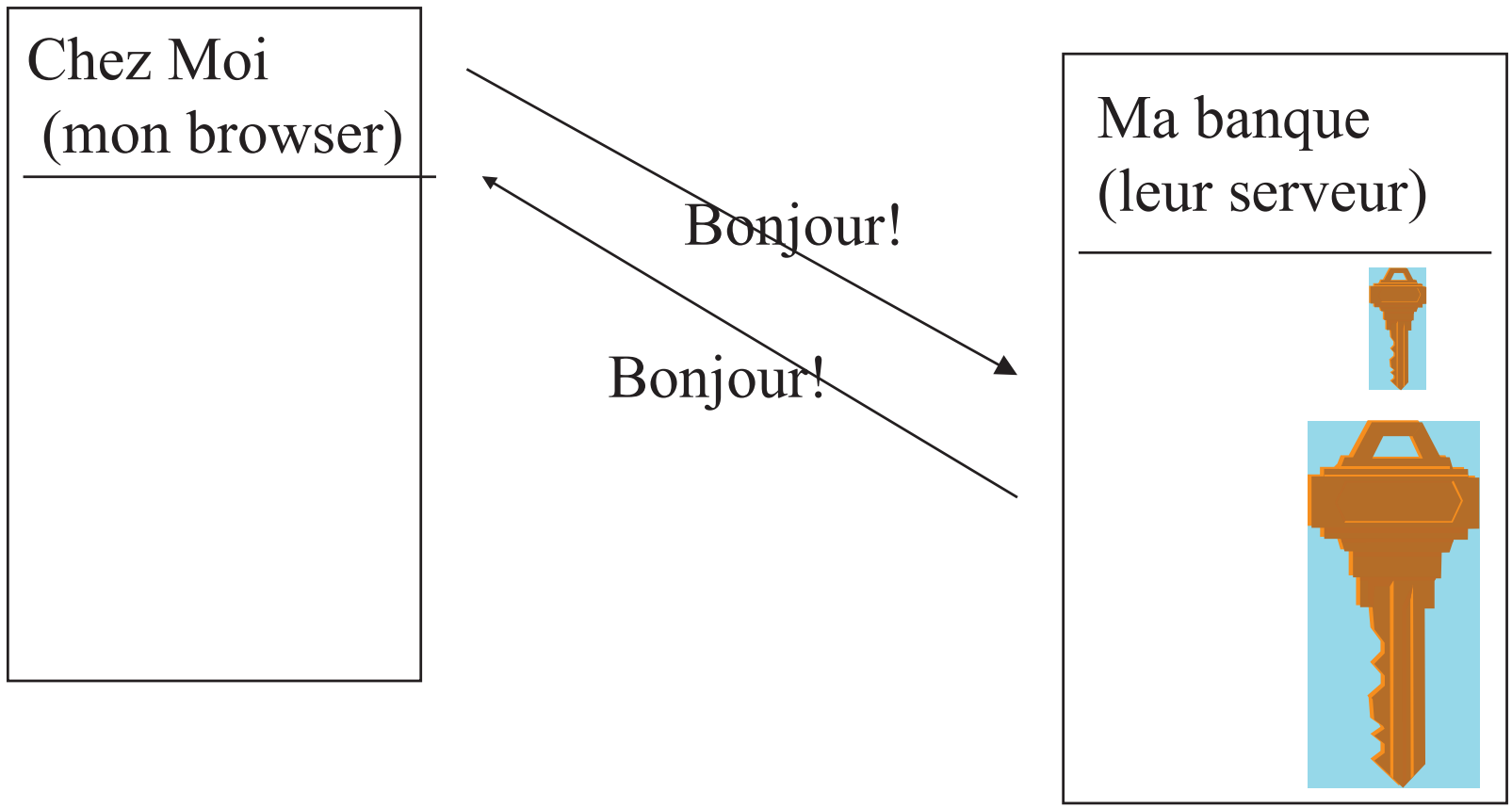
Chez Moi
(mon browser)

Ma banque
(leur serveur)



SSL

1) Connexion



SSL

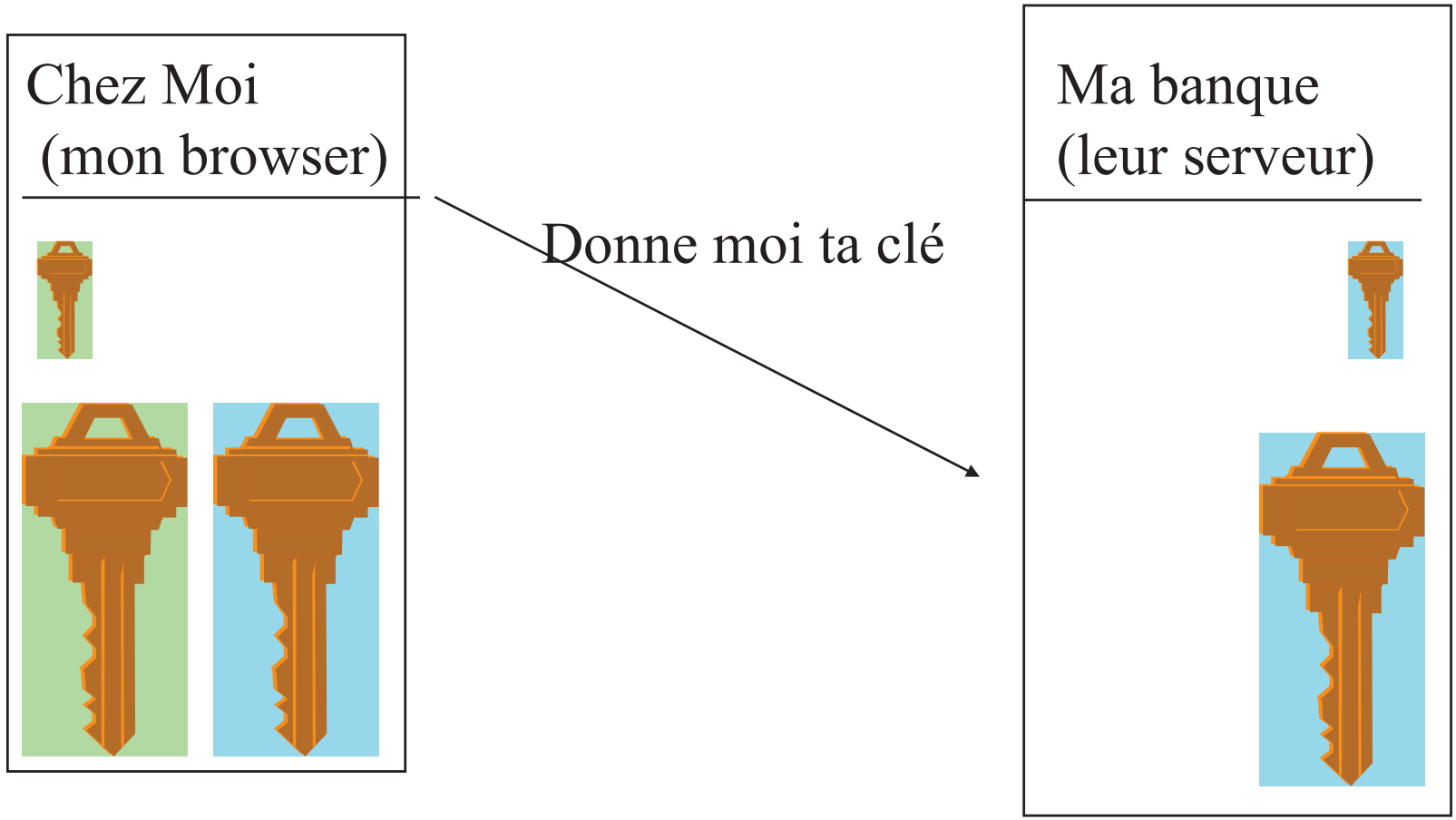
1) Génération de mes clés publique et privée

Chez Moi
(mon browser)

Ma banque
(leur serveur)

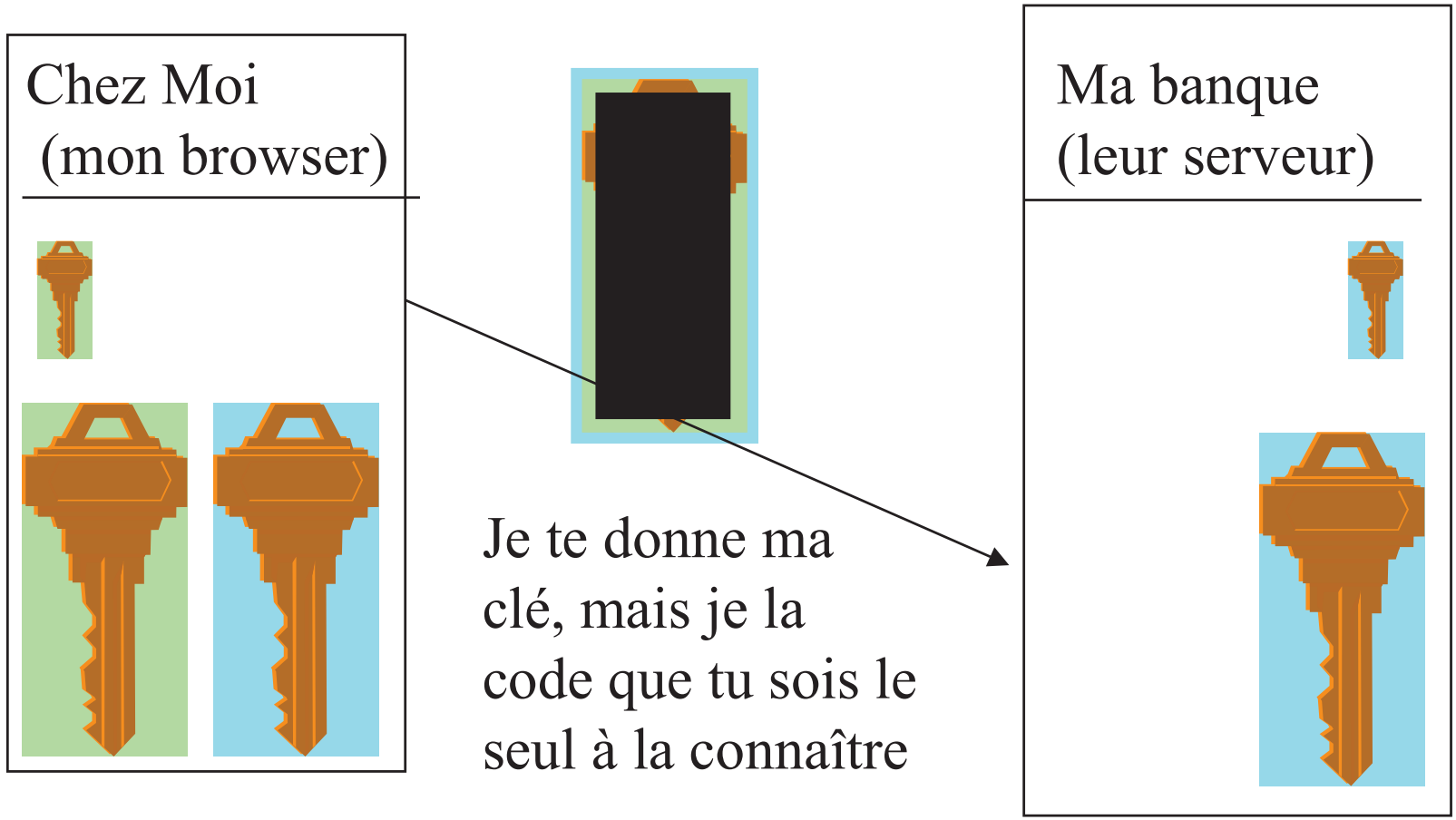
SSL

2) demande de la clé publique du serveur



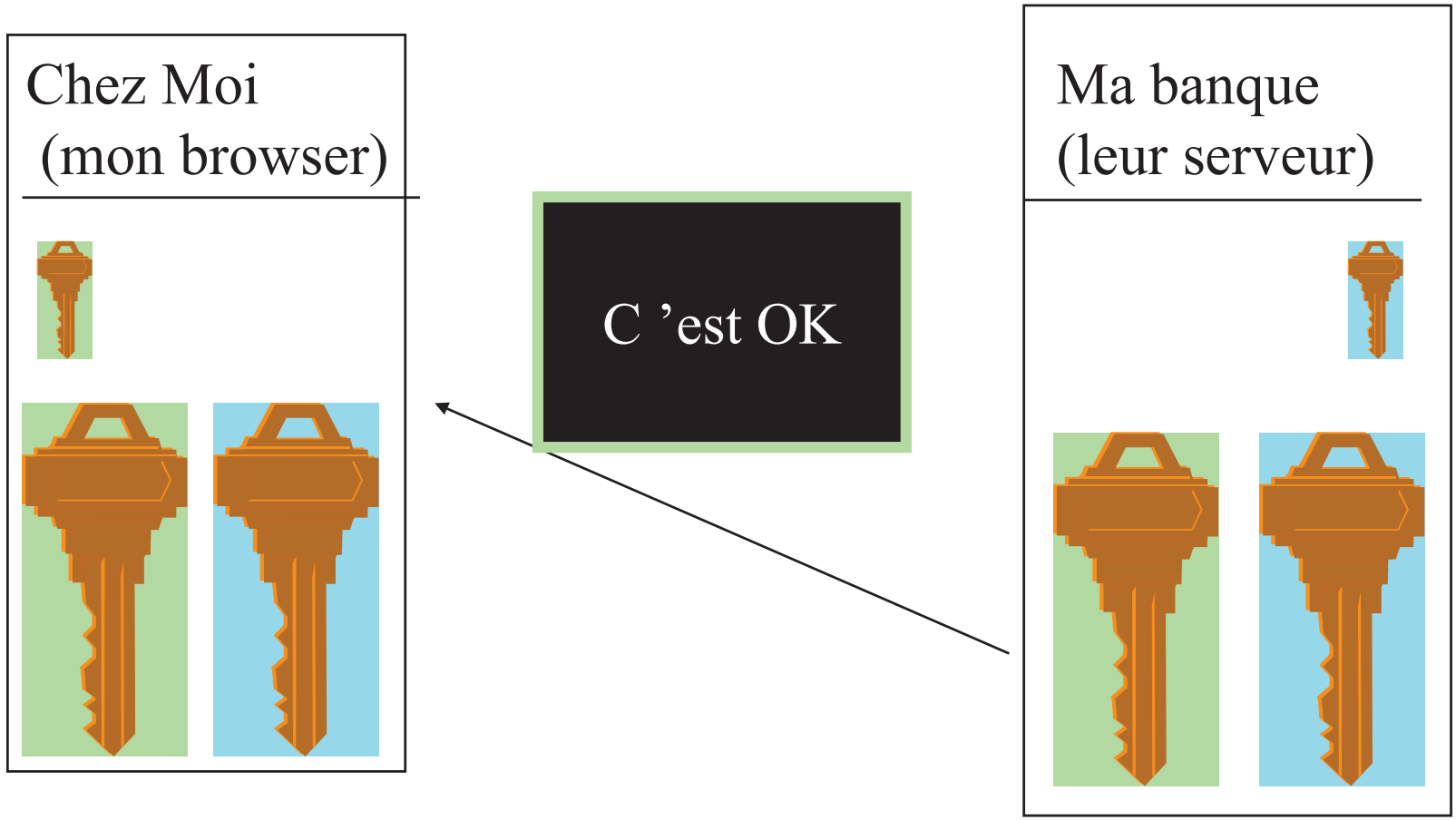
SSL

3) Encodage de ma clé publique



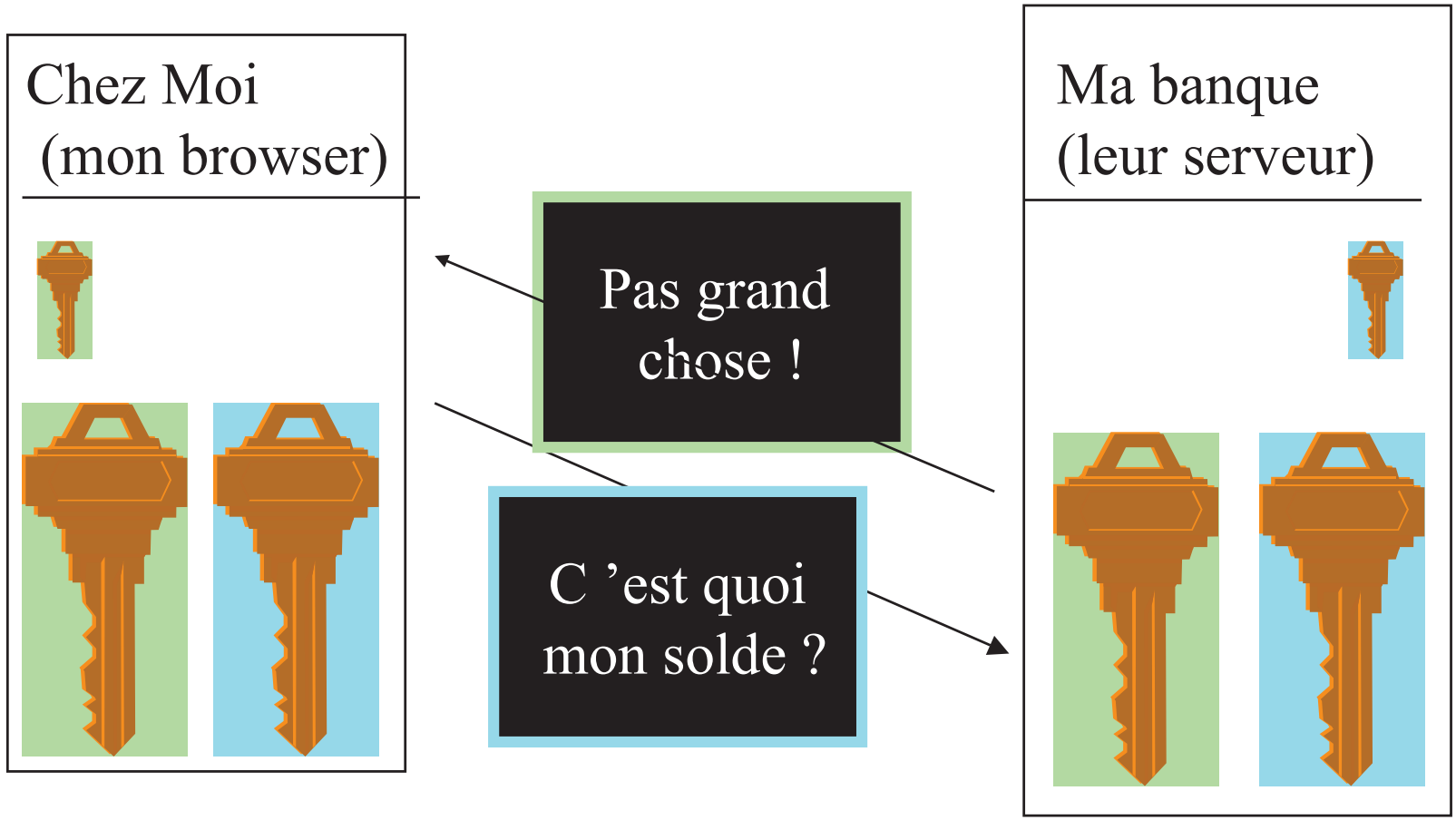
SSL

5) confirmation que tout c ’est bien passé



SSL

6) Echange d ’information sécurisé



Protocoles de sécurisation WEB

- Transport Layer Security (TLS)
- Secure Sockets Layer (SSL), prédécesseur de TSL

Services assurés : l'authentification du serveur, la confidentialité, l'intégrité ainsi que *la spontanéité*

spontanéité tout client peut se connecter de façon transparente à un serveur auquel il se connecte *pour la première fois*

Codage réalisée à la fois par

- cryptographie asymétrique, pour authentification et l'échanges des clés symétriques
- cryptographie symétrique (plus léger à réaliser) pour la transmission des informations

Session Sécurisée Client Web-Serveur Web 1

Protocoles de sécurisation WEB

- Transport Layer Security (TLS)
- Secure Sockets Layer (SSL), prédécesseur de TSL

Services assurés : l'authentification du serveur, la confidentialité, l'intégrité ainsi que *la spontanéité*

spontanéité tout client peut se connecter de façon transparente à un serveur auquel il se connecte *pour la première fois*

Codage réalisée à la fois par

- cryptographie asymétrique, pour authentification et l'échanges des clés symétriques
- cryptographie symétrique (plus léger à réaliser) pour la transmission des informations

Session Sécurisée Client Web-Serveur Web 2

Échange de clés privées de session

Plusieurs messages sont envoyé au début (phase “TSL Handshake”) :

- Messages *Serveur→Client* : codés avec la clé privée du serveur
- Messages *Client→Serveur* : codés avec la clé privée du client

Pour décoder les messages ci-dessus, le client a besoin de la clé publique du serveur et vice-versa

Authentifier la clé publique d’un serveur

- le serveur envoie au client un *certificat de sécurité* (clé publique serveur+identité serveur+ . . .)
- ce certificat est codé avec la clé privé d’une Autorité de Certification (AC)
- le client peut décoder le certificat avec la clé publique de l’AC, connue par tout navigateur

Session Sécurisée Client Web-Serveur Web 2

Échange de clés privées de session

Plusieurs messages sont envoyé au début (phase “TSL Handshake”) :

- Messages *Serveur*$\rightarrow$*Client* : codés avec la clé privée du serveur
- Messages *Client*$\rightarrow$*Serveur* : codés avec la clé privée du client

Pour décoder les messages ci-dessus, le client a besoin de la clé publique du serveur et vice-versa

Authentifier la clé publique d’un serveur

- le serveur envoie au client un *certificat de sécurité* (clé publique serveur+identité serveur+ . . .)
- ce certificat est codé avec la clé privé d’une Autorité de Certification (AC)
- le client peut décoder le certificat avec la clé publique de l’AC, connue par tout navigateur

Et avec le protocole *ssh*

Il n'y a plus d'Autorité de Certification (gestionnaire de clé publique)

Authentifier la clé publique du serveur

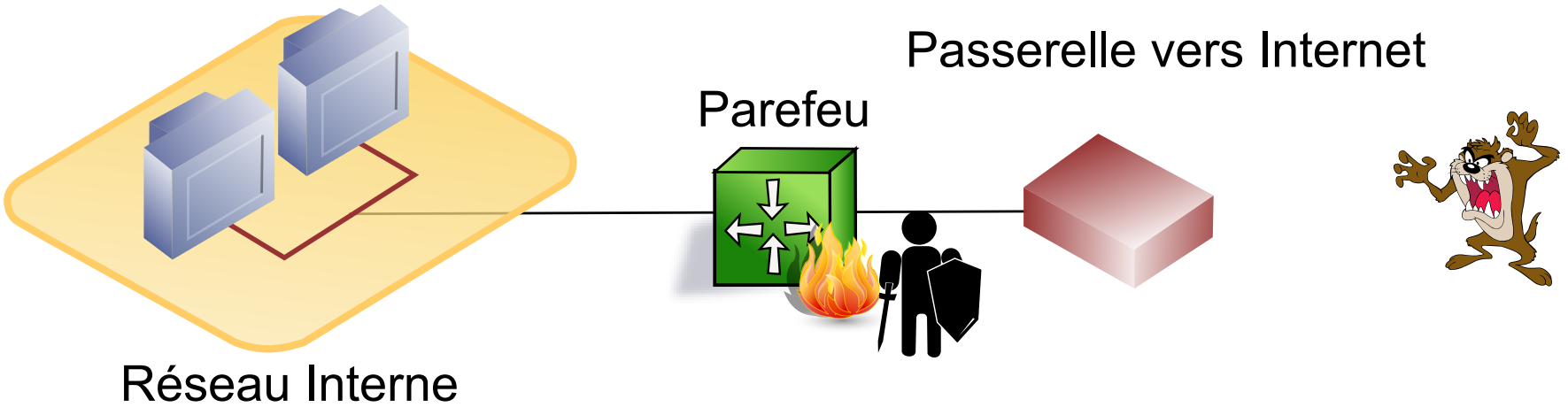
- Lors d'une première connexion, un message de confirmation est envoyé au client
- Le client est demandé de faire confiance à une clé publique (son résumé est affiché)

```
The authenticity of host 'iut-rt ' can't be establishe
RSA key fingerprint is 6f:e1:07:36:42:b2:0b:36:90:67:3
Are you sure you want to continue connecting (yes/no)?
```

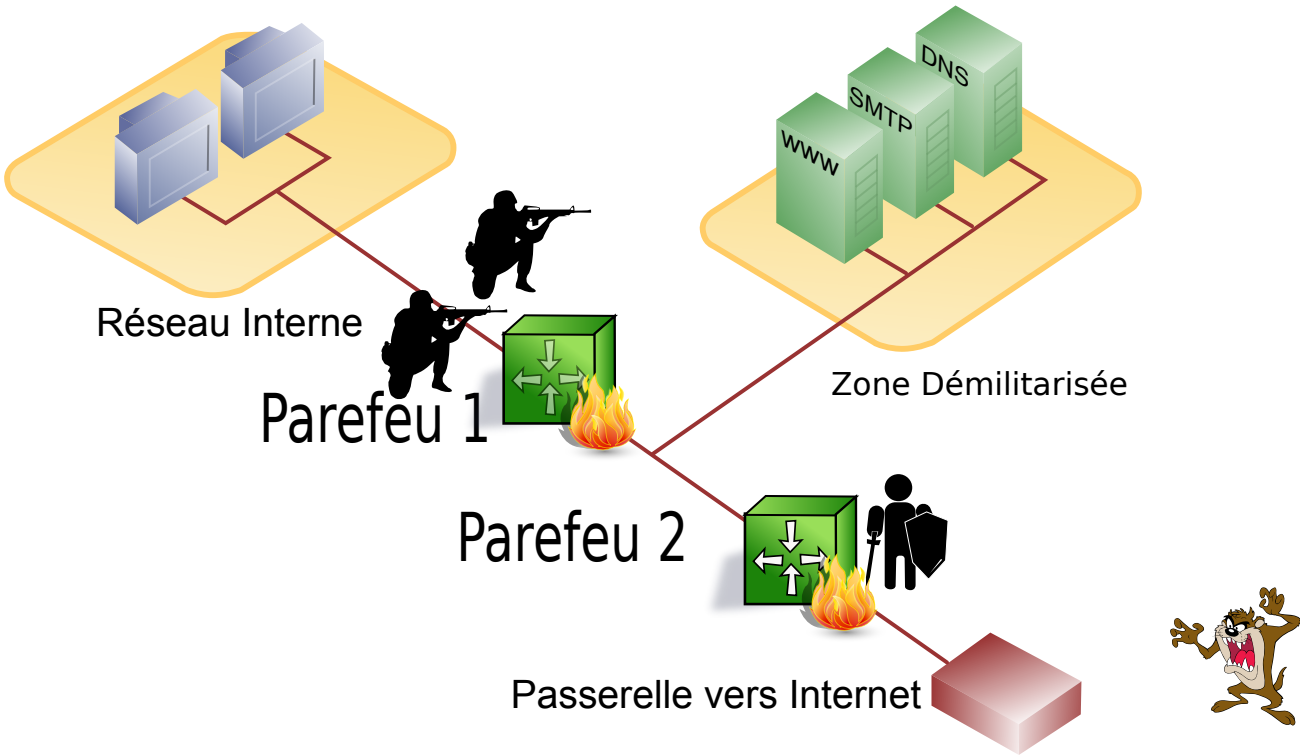
3. Solutions pratique de protection
   - Éléments de Cryptographie
   - Routage et filtrage via `Iptables`

Les pares-feu : 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`)



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

Filtrage de paquets : les règles

Les règles permettent d'identifier les paquets en fonction de :

- L'adresse IP source et l'IP destination
- Le port TCP, UDP source et/ou destination
- Le type de segment TCP
  - drapeau SYN activé $\Rightarrow$ ouverture de connexion
  - drapeau RST activé $\Rightarrow$ phase de fermeture de connexion
- L'interface sur laquelle le paquet arrive (`eth0`, `eth1`, `wlan0`)

Exemple de filtrage par adresse IP

Autoriser uniquement le trafic 10.0.0.9$\longleftrightarrow$le réseau local

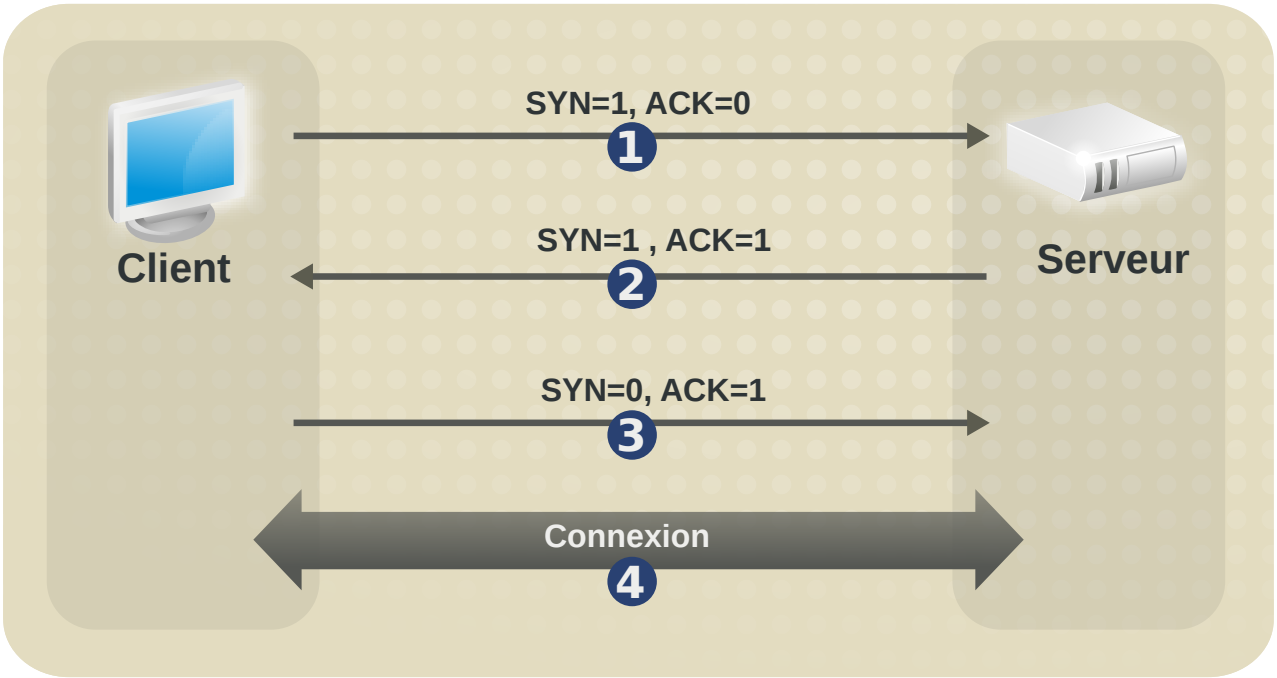
Règle	@ source	@destination	Ack = 1	Action
A	10.0.0.9	réseau local	-	Ok
B	réseau local	10.0.0.9	-	Ok
C	Toutes	Toutes	-	Refus

Filtrage plus fin à base d'analyse TCP 1/2

- TCP = protocole en mode connecté $\Rightarrow$ toute communication démarre avec l'ouverture de connexion
- Une politique de sécurité : autoriser uniquement les connexions initiées par une machine locale
  - Ouverture de connexion : le premier segment `SYN` envoyé a uniquement le drapeau `SYN` activé,
    - c'est le seul segment sans drapeau `ACK`

Filtrage plus fin à base d'analyse TCP 1/2

- TCP = protocole en mode connecté ⇒ toute communication démarre avec l'ouverture de connexion
- Une politique de sécurité : autoriser uniquement les connexions initiées par une machine locale
  - Ouverture de connexion : le premier segment `SYN` envoyé a uniquement le drapeau `SYN` activé,
    - c'est le seul segment sans drapeau `ACK`



Filtrage plus fin à base d'analyse TCP 2/2

Autoriser uniquement des connections vers l'extérieur via Telnet

	Règle	IP Src	IP Dst	Port Src	Port Dest	Ack	Action
	A	Locale	Toutes	> 1023	23	Oui/Non	Ok
↓	B	Toutes	Locale	23	> 1023	Oui	Ok
	C	Toutes	Locale	23	> 1023	Non	Refus
	D	Toutes	Toutes	Tous	Tous	Oui/Non	Refus

Règle A pour les paquets envoyés vers l'extérieur

- IP source : une adresse locale
- port destination : **23**, port source : **>1023**
- le premier paquet (qui établit la connexion) à le bit **ACK = 0**, les autres l'ont à **1** $\Rightarrow$ les deux cas sont **autorisés**

Règles B,C pour les paquets venant de l'extérieur

- port source : **23**, port destination : supérieur à **1023**
- tous les paquets auront le bit **ACK = 1**$\Rightarrow$ les autres sont rejetés : **ACK = 0**

Filtrage plus fin à base d'analyse TCP 2/2

Autoriser uniquement des connections vers l'extérieur via Telnet

	Règle	IP Src	IP Dst	Port Src	Port Dest	Ack	Action
	A	Locale	Toutes	$>$ 1023	23	Oui/Non	Ok
	B	Toutes	Locale	23	$>$ 1023	Oui	Ok
	C	Toutes	Locale	23	$>$ 1023	Non	Refus
	D	Toutes	Toutes	Tous	Tous	Oui/Non	Refus

Règle A pour les paquets envoyés vers l'extérieur

- IP source : une adresse locale
- port destination : **23**, port source : **>1023**
- le premier paquet (qui établit la connexion) à le bit **ACK = 0**, les autres l'ont à **1** $\Rightarrow$ les deux cas sont **autorisés**

Règles B,C pour les paquets venant de l'extérieur

- port source : **23**, port destination : supérieur à **1023**
- tous les paquets auront le bit **ACK = 1**$\Rightarrow$ les autres sont rejetés : **ACK = 0**

Comment fonctionne un routeur ?

- Chaque paquet IP reçu est examiné pour savoir si il est destiné à la machine locale ou pas ⇒ Décision de routage :
  A. S'il est destiné à la machine locale, il est traité localement
  B. Sinon, la table de routage est interrogée pour déterminer la route

Le Filtrage

Le filtrage peut intervenir avant ou après la décision de routage

- Exemple : si le port destination est modifié avant la décision de routage, la décision de routage peut être modifiée

Comment fonctionne un routeur ?

- Chaque paquet IP reçu est examiné pour savoir si il est destiné à la machine locale ou pas ⇒ Décision de routage :
  - A S'il est destiné à la machine locale, il est traité localement
  - B Sinon, la table de routage est interrogée pour déterminer la route

Le Filtrage

Le filtrage peut intervenir avant ou après la décision de routage

- Exemple : si le port destination est modifié avant la décision de routage, la décision de routage peut être modifiée

Comment fonctionne un routeur ?

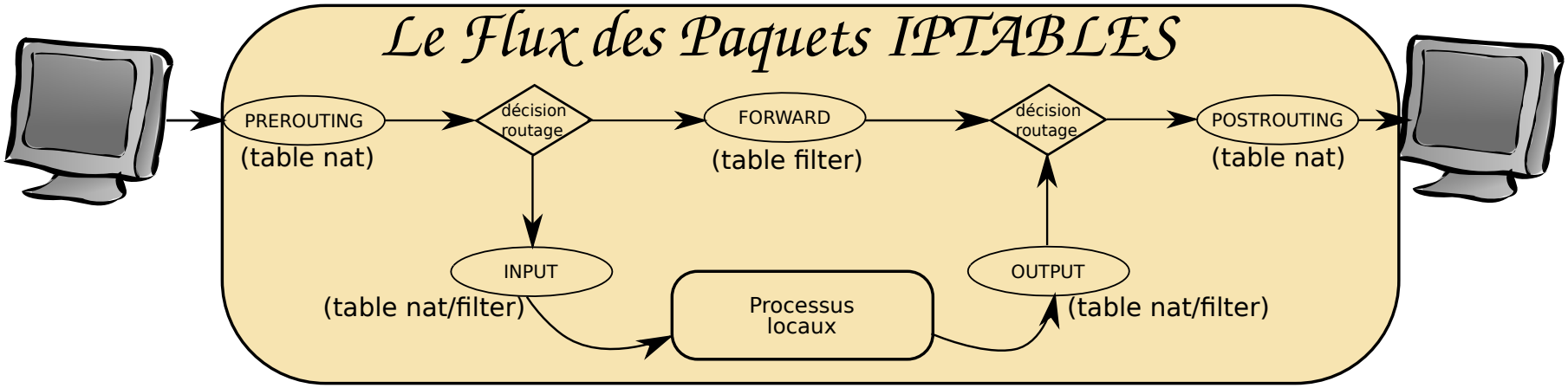
- Chaque paquet IP reçu est examiné pour savoir si il est destiné à la machine locale ou pas ⇒ Décision de routage :
  - A S'il est destiné à la machine locale, il est traité localement
  - B Sinon, la table de routage est interrogée pour déterminer la route

Le Filtrage

Le filtrage peut intervenir avant ou après la décision de routage

- Exemple : si le port destination est modifié avant la décision de routage, la décision de routage peut être modifiée

Le filtrage sous Linux : iptables



- Les règles sont rangées dans des chaînes de règles et appliquées une par une
- Plusieurs chaînes de règles par défaut : `PREROUTING`, `INPUT`, `OUTPUT`, `FORWARD`, `POSTROUTING`
- La commande `iptables` permet de manipuler les chaines
  - liste les règles d'une chaîne (`iptables -L`), vide la chaîne (`iptables -F`), fixe la politique d'une chaîne (`iptables -P`)
- Exemples :

```
iptables -L INPUT
iptables -P OUTPUT DROP
iptables -P FORWARD DROP
```

La manipulation des règles d'une chaîne

- Ajouter une règle à la fin (A)
- Insertion d'une règle au début (I)
- Modification d'une règle (R)
- Suppression d'une règle (D)
- Exemples :

```
iptables -I INPUT -s 213.186.33.2 -j DROP
iptables -A INPUT -s 213.186.33.2 -j ACCEPT
iptables -I INPUT -s 213.186.33.2 -j ACCEPT
iptables -D INPUT 1
iptables -F INPUT
```

L'action d'une règle `iptables`

- `DROP` : supprimer le paquet
- `REJECT` : supprimer le paquet et envoyer un message d'erreur
- `ACCEPT` : accepter le paquet
- `DNAT` et `SNAT` : changer la source ou la destination du paquet

Syntaxe générique d'une règle

```
iptables [-t table] -A/I/D chaine [cond] [action]
```

où `cond` indique une condition :

- IP Source (`-s 192.168.37.0/24`)
- Destination (`-d 192.168.37.9`)
- Protocole (`-p TCP`)
- Le port source (`-sport 22`) et destination (`-dport 80`), etc

et `table` est

- `nat` : table utilisée pour faire de la translation d'adresse
- `filter` : table par défaut, permet les actions DROP/REJECT
- `mangle` : tablée utilisée pour marquer les paquets