Compléments sur les réseaux et les applications réparties

Renaud Lachaize
Université Grenoble Alpes

Plan

- Exemples de protocoles applicatifs
  - FTP : transfert de fichiers
  - Courrier électronique
    - SMTP
    - POP3 / IMAP
    - Interface web pour la messagerie électronique
- Sécurité
  - Principaux types de menaces
  - Principes des moyens de protection
  - Exemples d'outils pour la sécurité

FTP : File Transfer Protocol
Principes

- Rôle : Transfert de fichier(s) depuis/vers une machine distante
- Service basé sur TCP
  - Serveur (démon) : écoute sur le port 21 par défaut
  - Client (ligne de commande ou graphique) : numéro de port alloué dynamiquement
- Similarités avec HTTP
  - Messages de contrôle échangés sous forme de texte ASCII
- Différences avec HTTP
  - Canaux de communications différents pour les message de contrôle (requêtes/réponses) et les données (fichiers à transférer).
    - On dit parfois que le contrôle est effectué hors-bande (*out-of-band*).
  - Le serveur doit conserver des informations sur l'état d'un client pendant toute la durée d'une connexion

FTP : File Transfer Protocol
Principes (Suite)

- Connexion de contrôle
  - Établie à l'initiative du client
  - Véhicule les requêtes du client et les acquittements du serveur
- Connexion de données
  - Utilisée pour transférer des informations telles que le contenu d'un fichier ou la liste des fichiers disponibles dans un répertoire
  - Deux modes d'interaction possibles :
    - Mode (serveur) **actif** : le client indique au serveur le numéro de port à contacter (côté client) et le serveur établit la connexion
    - Mode (serveur) **passif** : le serveur indique au client le numéro de port à contacter (côté serveur) et le client établit la connexion
  - La connexion ouverte pour le transfert de données est généralement fermée à l'issue de chaque transfert (mais la connexion de contrôle est maintenue)
  - Par défaut, on utilise le port 20 (côté serveur) pour les échanges de données

FTP : File Transfer Protocol
Principes (Suite)

- Etat associé à une connexion de contrôle
  - Conservé par le serveur pour chaque client
  - Mémorisation de l'identité (nom d'utilisateur, pour vérifier les droits d'accès) et du répertoire courant
- Informations de contrôle
  - Codées en ASCII 7-bit
  - Commandes : sur 4 caractères suivies d'éventuels arguments, terminaison par <CR><LF>
  - Acquittement systématique, avec code sur 3 chiffres
- Deux formats principaux pour l'échange de données
  - Binaire : le contenu du fichier est transmis tel quel
  - ASCII : le fichier est interprété comme un fichier texte, avec d'éventuelles conversions pour le codage de certains caractères (fin de ligne notamment)
    - À utiliser avec précaution (dans le doute, utiliser le mode binaire)

FTP : File Transfer Protocol
Exemple (mode actif)

```
Client :                                                    Serveur :
USER john
                                    331 (username OK, password required)
PASS mdp1234
                                                230 (user logged in, proceed)
LIST
                                150 (OK, about to open data connection)
                    226 (requested action OK, closing data connection)
CWD my_directory
                                                        250 (CWD successful)
RETR doc.txt
                                150 (OK, about to open data connection)
                    226 (requested action OK, closing data connection)
QUIT
                                        221 (closing control connection)
```

FTP : File Transfer Protocol
Compléments

- Limites
  - Métadonnées : les attributs de datation d'un fichier ne sont pas préservés lors d'un transfert
  - Sécurité : toutes les informations sont transmises en clair, sans aucune garantie de confidentialité
  - Performances : latence importante (cf. nombre de commandes et de connexions nécessaires pour un ou plusieurs transferts)

- Spécification détaillée
  - Voir RFC 959 (1985) : http://tools.ietf.org/html/rfc959

- Améliorations (au niveau de la sécurité)
  - FTPS : Utilisation du cryptage au niveau des sockets (SSL/TLS)
  - FTP/SSH : nouveau protocole (intégration avec SSH), voir plus loin

Courrier électronique
Introduction

- L'une des applications les plus populaires de l'Internet depuis ses débuts

- Moyen de communication asynchrone
  - À l'instar du courrier postal
  - Les moments associés à la rédaction, à l'envoi, à la réception et à la consultation des messages peuvent être espacés dans le temps, en fonction des contraintes des intervenants

- Envois en masse faciles et rapides
  - Envoi d'un même message à de nombreux destinataires
  - À bon ou mauvais escient (listes de diffusion / spam)

Courrier électronique
Principaux concepts

- Agent utilisateur (*mail user agent*)
  - Programme utilisé par une personne pour consulter, trier, stocker, rédiger, envoyer (…) ses messages. L'interface d'accès peut varier (ligne de commande ou interface graphique).
  - On parle parfois de *client de messagerie* (*email client*).
- Agent de transfert (*mail transfer agent*)
  - Egalement appelé serveur de messagerie (*mail server*).
  - Programme chargé de l'envoi et de la réception du courrier pour un ensemble d'utilisateurs. Fonction analogue à celle d'un bureau de poste.
- Protocoles de messagerie
  - Envoi et acheminement de messages : SMTP, webmail
  - Récupération, consultation de messages : POP3, IMAP, webmail
  - Tous ces protocoles applicatifs sont basés sur TCP

Courrier électronique
SMTP : Simple Mail Transfer Protocol

- Protocole d'acheminement de messages
  - Défini dans le RFC 2821 : http://tools.ietf.org/html/rfc2821
  - Spécification initiale : 1982 (et existence antérieure à cette date)
  - La consultation des boîtes aux lettres n'est pas gérée par SMTP
- Principe de base
  - Alice rédige un message et indique l'adresse de Bob pour le destinataire
  - L'agent utilisateur d'Alice (ici client SMTP) envoie le message d'Alice à son agent de transfert (ici serveur SMTP), où il est placé dans une file d'émission
  - L'agent de transfert d'Alice agit alors comme client SMTP : il récupère le message et établit une connexion TCP avec l'agent de transfert de Bob (ici serveur SMTP)
  - Après un dialogue d'initialisation, l'agent de transfert d'Alice (ici client SMTP) transmet le message à l'agent de transfert de Bob (ici serveur SMTP) via la connexion TCP
  - L'agent de transfert de Bob stocke le message reçu dans la boîte aux lettres de Bob (typiquement un fichier local sur le serveur)

Courrier électronique
SMTP : Simple Mail Transfer Protocol

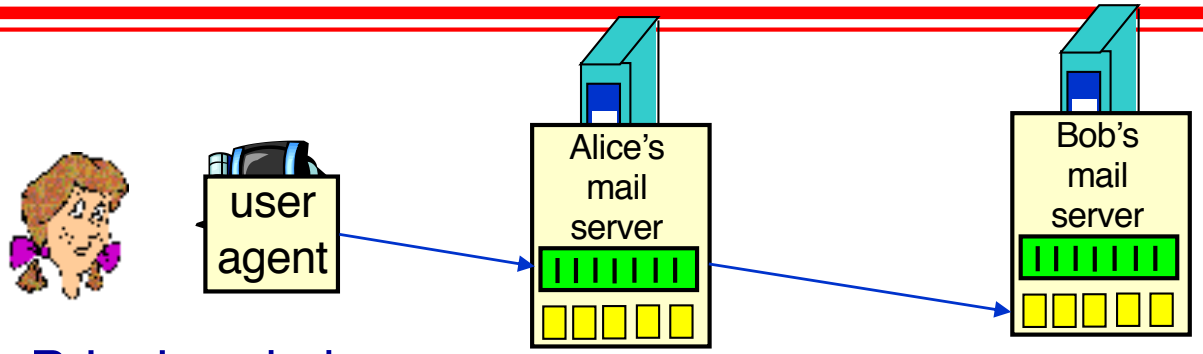


Schéma tiré de Kurose & Ross, *Computer Networking*

- Principe de base
  - Alice rédige un message et indique l'adresse de Bob pour le destinataire
  - L'agent utilisateur d'Alice (ici client SMTP) envoie le message d'Alice à son agent de transfert (ici serveur SMTP), où il est placé dans une file d'émission
  - L'agent de transfert d'Alice agit alors comme client SMTP : il récupère le message et établit une connexion TCP avec l'agent de transfert de Bob (ici serveur SMTP)
  - Après un dialogue d'initialisation, l'agent de transfert d'Alice (ici client SMTP) transmet le message à l'agent de transfert de Bob (ici serveur SMTP) via la connexion TCP
  - L'agent de transfert de Bob stocke le message reçu dans la boîte aux lettres de Bob (typiquement un fichier local sur le serveur)

Courrier électronique
Compléments sur SMTP

- On peut avoir une communication directe entre le client SMTP de l'émetteur et le serveur SMTP du récepteur
  - Le courrier électronique, découpé en paquets, peut transiter par de nombreux routeurs mais, au niveau applicatif (SMTP), il n'y a pas nécessairement d'acteur intermédiaire entre le client et le serveur
  - Si le serveur n'est pas joignable, le client SMTP conserve le message et tente de le réémettre périodiquement (jusqu'à un seuil max => abandon). L'émetteur (Alice) n'a pas à se préoccuper du problème et peut se déconnecter.
- Cependant, on utilise dans certains cas des relais/passerelles de messagerie (*email gateways*)
  - Correspondance avec la structure arborescente d'une organisation
  - Equilibrage de charge (plusieurs serveurs frontaux qui aiguillent les messages en fonction du destinataire)
  - Redirection d'adresse
  - Avantage : transparence et simplicité du point de vue des émetteurs (il suffit de connaître l'adresse du destinataire)

Courrier éléctronique
Compléments sur SMTP

- On peut voir la liste des clients/serveurs SMTP utilisés pour l'acheminement d'un message en consultant le(s) champ(s) **`Received:`** de l'en-tête d'un message

```
Received: from mx-serv.inrialpes.fr (mx-serv.inrialpes.fr [194.199.18.100])
          by dwimmerlaik.inrialpes.fr (8.13.6/8.11.3/ImagV2) with ESMTP id m2LEehQI022923
          for <rlachaiz@dwimmerlaik.inrialpes.fr>; Fri, 21 Mar 2008 15:40:43 +0100 (MET)

Received: from mail2-relais-roc.national.inria.fr (mail2-relais-roc.national.inria.fr
[192.134.164.83])
          by mx-serv.inrialpes.fr (8.13.6/8.13.0) with ESMTP id m2LEedwk025267
          for <Renaud.Lachaize@inrialpes.fr>; Fri, 21 Mar 2008 15:40:40 +0100 (MET)

Received: from imag.imag.fr ([129.88.30.1])
  by mail2-smtp-roc.national.inria.fr with ESMTP; 21 Mar 2008 15:40:39 +0100

Received: from smtp38.bv.rhone-alpes.fr (nat38.bv.rhone-alpes.fr [193.54.133.237])
          by imag.imag.fr (8.13.8/8.13.8) with ESMTP id m2LEdMTb021548
          for <renaud.lachaize@imag.fr>; Fri, 21 Mar 2008 15:39:23 +0100 (CET)

Received: from smtp38.bv.rhone-alpes.fr ([192.168.55.11]) by smtp38.bv.rhone-alpes.fr with
Microsoft SMTPSVC(6.0.3790.1830); Fri, 21 Mar 2008 15:39:17 +0100
```

À lire de bas en haut (chemin de l'émetteur vers le récepteur)

Courrier électronique
Lien entre SMTP et DNS

- Comment un client SMTP trouve-t-il l'adresse IP d'un serveur SMTP à contacter à partir de l'adresse email du destinataire ?
- Réponse : on peut utiliser le DNS
  - … mais cela pose un problème. Par exemple, pour les adresses @microsoft.com, cela impose que la(les) machine(s) correspondant à ce nom (microsoft.com) héberge(nt) à la fois un serveur web (pour l'URL http://microsoft.com) et un serveur SMTP.
    - C'est possible mais peu satisfaisant en termes d'administration, d'efficacité et de sécurité.
  - Pour régler ce problème, on pourrait imposer une convention. Par exemple, chaque organisation pourrait préfixer le nom de son serveur de messagerie par smtp, mail ou autre chose.
    - Problème : les adresses deviendraient moins simples à utiliser (e.g. @mail.microsoft.com) mais c'est envisageable (cf. préfixe courant www pour les serveurs Web).
  - Compromis : on utilise un type d'entrée particulier (MX) dans le DNS pour indiquer/connaître le nom et l'adresse du ou des serveurs de messagerie associés à un domaine.

Courrier électronique
Lien entre SMTP et DNS (suite)

- Exemple :

```
$ dig MX microsoft.com

;; QUESTION SECTION:
;microsoft.com.                 IN      MX

;; ANSWER SECTION:
microsoft.com.          1735    IN      MX      10 mailc.microsoft.com.
microsoft.com.          1735    IN      MX      10 maila.microsoft.com.
microsoft.com.          1735    IN      MX      10 mailb.microsoft.com.

;; ADDITIONAL SECTION:
mailb.microsoft.com.    1735    IN      A       205.248.106.30
mailc.microsoft.com.    1735    IN      A       131.107.115.214
maila.microsoft.com.    1735    IN      A       205.248.106.64
mailc.microsoft.com.    1735    IN      A       205.248.106.32
mailb.microsoft.com.    1735    IN      A       131.107.115.215
maila.microsoft.com.    1735    IN      A       131.107.115.212
```

Courrier électronique
SMTP : dialogue entre client et serveur

- Requêtes et réponses au format ASCII 7 bit
- Marqueur <CR><LF> à la fin de chaque ligne
- Possibilité d'envoyer plusieurs messages via une même connexion
- Exemple (tiré de L.Peterson & B. Davie, *Computer Networks: a systems approach*)

```
Client (cs.princeton.edu) :                                  Serveur (cisco.com) :
HELO  cs.princeton.edu
                                  250 Hello daemon@mail.cs.princeton.edu (128.12.169.24)
MAIL FROM: bob@cs.princeton.edu
                                                                          250 OK
RCPT TO: <alice@cisco.com>
                                                                          250 OK
RCPT TO: <tom@cisco.com>
                                                                 550 No such user here
DATA
                                          354 Start mail input; end with <CRLF>.<CRLF>
From: bob@cs.princeton.edu
To: alice@cisco.com
... (autres champs tels que Date, Subject, etc.)
      <-- séparateur entre en-tête et corps du message
Hello, blah blah blah ...
.     <-- marqueur de fin de mesage
                                                                          250 OK
QUIT
                                                               221 Closing connection
```

dialogue préalable à l'envoi du message

SMTP : Format des messages

- Pour des raisons historiques, l'intégralité d'un message est encodée en ASCII (en-tête, corps, fichiers joints)

- En-tête
  - Nom de l'émetteur, du destinataire, sujet, etc.
  - Ne correspond pas aux commandes MAIL FROM et RCPT TO du dialogue SMTP. L'en-tête fait partie du message (DATA).
  - Séparé du corps du message par une ligne vide.

- Corps du message
  - Problème : comment peut-on insérer des caractères non ASCII ou des fichiers multimédia dans un message ?
  - Solution : MIME (Multipurpose Internet Mail Extensions) : En-têtes supplémentaires qui précisent
    - le type de contenu (*Content-Type:*)
    - leur encodage (*Content-Transfer-Encoding:*)

Courrier électronique
SMTP : comparaison avec HTTP

- Un envoi de message via SMTP correspond à un mode d'interaction *push*
  - Push : la connexion est à l'initiative de l'émetteur des données
  - Pull : la connexion est à l'initiative du récepteur des données
  - HTTP GET : pull ; HTTP PUT : push

- Contrairement à HTTP
  - Il n'est pas possible de transférer directement des données binaires. Il est nécessaire de les ré-encoder sous forme de caractères ASCII 7 bits.
  - Pour le transfert d'un document composite (texte HTML + images), SMTP n'utilise qu'un seul message.

Courrier électronique
Protocoles d'accès aux messages

- Un protocole d'envoi de messages (tel que SMTP) n'est pas suffisant
  - Un serveur SMTP doit être toujours joignable (et bien administré)
  - Les utilisateurs ne sont pas toujours connectés
  - Les utilisateurs préfèrent en général consulter/gérer leur courrier depuis leur propre machine plutôt qu'en se connectant à distance sur un serveur
  - Conséquence : Besoin d'un protocole complémentaire pour permettre à un utilisateur (destinataire) de consulter sa boîte aux lettres quand il le souhaite (protocole de type *pull*) via son client de messagerie

- Deux principaux protocoles d'accès aux messages
  - Post Office Protocol v3 (POP3)
  - Internet Mail Access Protocol (IMAP)

Courrier électronique
POP3

- Principales caractéristiques
  - Protocole simple et relativement limité
  - Requêtes/réponses au format ASCII
  - Utilise par défaut le port 110 sur le serveur
  - Défini dans le RFC 1939 : http://tools.ietf.org/html/rfc1939
- Trois phases :
  - Autorisation (*authorization*) : envoi des identifiants (en clair, par défaut)
  - Transaction : récupération des messages (complets) et marquage éventuel (par le client) des messages à supprimer
  - Mise à jour (*update*) : suppression des messages marqués sur le serveur et fermeture de la connexion
- Nécessité de conserver, sur le serveur, un état du dialogue avec un client
  - Pour la durée d'une connexion
  - ... mais pas au delà

Courrier eléctronique
POP3 : Exemple

Source : Kurose & Ross, *Computer Networking*

authorization phase

- client commands:
  - **`user:`** declare username
  - **`pass:`** password
- server responses
  - **`+OK`**
  - **`-ERR`**

transaction phase, client:

- **`list:`** list message numbers
- **`retr:`** retrieve message by number
- **`dele:`** delete
- **`quit`**

```
S: +OK POP3 server ready
C: user bob
S: +OK
C: pass hungry
S: +OK user successfully logged on
```

```
C: list
S: 1 498
S: 2 912
S: .
C: retr 1
S: <message 1 contents>
S: .
C: dele 1
C: retr 2
S: <message 1 contents>
S: .
C: dele 2
C: quit
S: +OK POP3 server signing off
```

Courrier électronique
IMAP

- Restrictions de POP3
  - Impossibilité d'organiser les messages laissés sur le serveur (dossiers)
  - Peu convenable pour les utilisateurs nomades (plusieurs machines)
- Gestion des messages sur le serveur avec IMAP
  - Chaque message est associé à un dossier (Inbox par défaut)
  - Possibilité de créer des dossiers et de déplacer des messages
  - Possibilité de faire des recherches sur le contenu des messages
  - Possibilité de ne récupérer que les en-têtes des messages ou un sous-ensemble des fichiers joints
  - Synchronisation entre le client de messagerie et le serveur par le biais d'attributs associés aux messages
- Le serveur IMAP doit conserver des informations entre deux sessions (nom et contenu des dossiers)
- Protocole significativement plus complexe que POP3
- Port serveur 143 par défaut, commandes au format ASCII
- Défini dans le RFC 3501 : http://tools.ietf.org/html/rfc3501

Courrier électronique
Webmail

- Interface de consultation et d'envoi de messages via un navigateur web
- Utilisation du protocole HTTP pour les échanges entre le client et le serveur web
- Le serveur web génère des pages web dynamiques
  - Les liens présents sur les pages générées correspondent à des actions et pointent vers des programmes/scripts CGI présents sur le serveur Web
  - Ces programmes interagissent avec le serveur de messagerie via les protocoles vus précédemment et mettent en forme les informations reçues
    - Pour la consultation de messages, envoi de commandes POP3 ou IMAP
    - Pour l'envoi d'un message, envoi d'une requête SMTP

Sécurité des réseaux
Introduction

- Préoccupation majeure à l'heure de l'Internet
  - Ubiquité des applications réparties
  - Faiblesse inhérente des logiciels modernes en raison de leur complexité

- Idées reçues
  - Il faut être connu pour être la cible d'une attaque de sécurité
    - Non : Balayage automatique de plage d'adresses IP et de ports
  - Seul un expert peut attaquer une machine
    - Non : Il est (relativement) simple de trouver des bibliothèques d'attaques adaptées aux logiciels serveurs/clients rencontrés, prêtes à l'emploi
  - Les symptômes d'une attaque sont faciles à repérer
    - Non : de nombreuses attaques sont discrètes, silencieuses

Sécurité des réseaux
Principaux types de menaces

- Espionnage/interception des paquets sur un réseau
  - Récupération d'informations confidentielles par écoute passive d'un canal de communication (difficile à détecter)
  - Ajout/destruction de paquets, modification de leur contenu
- Détournement d'identité
  - Imitation de l'émetteur ou détournement de l'émetteur
  - Imitation du destinataire (attention au DNS)
- Déni de service d'une machine ou d'un réseau
  - Arrêt d'un service ou d'une machine grâce à l'exploitation d'une faille
  - Saturation (débit / demandes de connexions)
- Diffusion de logiciels malveillants (*malware*)
  - Diverses formes/modes de propagation : virus, vers, chevaux de troie
  - Palette d'effets très vastes : dysfonctionnements, vol/destruction d'informations, détournement d'identité, détournement de machines

Sécurité des réseaux
Principes des réponses aux grands types de menaces

- Détournement et exploitation de failles
  - Approche : Réduire la vulnérabilité d'une machine
  - Mises à jour de sécurité (correctifs pour les failles)
  - Outils de filtrage des communications/actions, outils de diagnostic
  - Cloisonnement des privilèges
  - Randomisation de l'espace mémoire
- Espionnage des informations
  - Approche : Rendre les informations inexploitables pour l'espion
  - Cryptographie : utilisation d'un code secret, connu seulement de l'émetteur et du destinataire final
- Usurpation d'identité, modification des (séquences de) messages
  - Approche : Authentification et vérifications d'intégrité
  - Un récepteur doit pouvoir établir avec certitude
    - qu'un message a bien été envoyé par l'émetteur annoncé
    - que le contenu d'un message n'a pas été modifié en chemin
    - …
  - Mots de passe, communications et signatures cryptographiques, etc.
- Déni de service par saturation
  - Approche : détection et filtrage anticipés des paquets/requêtes suspects

Sécurité des réseaux
Pare-feu (ou garde-barrière, *firewall*)

- Objectif : filtrage des communications entrantes et/ou sortantes au niveau d'une machine ou d'un réseau
- Implémentation logicielle et/ou matérielle
  - Compromis flexibilité/efficacité
- Définition de règles d'interdiction et d'autorisation de certains types de communication
- Filtrage applicable à plusieurs niveaux
  - Paquets (approche générique) :
    - Sans état : examen du protocole de transport (TCP/UDP), des adresses IP et des numéros de port de la source et de la destination
    - Avec état : suivi du statut des connexions
  - Niveau applicatif (approche spécialisée) : examen des messages propres à un protocole applicatif (HTTP, SMTP, …)

Principes élémentaires de cryptographie

- Objectif principal : préserver la confidentialité d'une communication même si les messages sont interceptés
- Moyen : transformations de messages à l'aide de clés (informations gardées secrètes)
- Deux principales classes d'algorithmes utilisées en pratique
  - Cryptographie symétrique : clé secrète partagée entre les deux parties
  - Cryptographie asymétrique : Paire clé publique / clé privée
    - On chiffre un message avec la clé publique du destinataire, que seul ce dernier pourra déchiffrer (avec sa clé privée)
    - Moins efficace mais permet de résoudre le problème de sécurité du canal de communication de la clé
  - Les deux méthodes sont généralement utilisées de façon combinée

Sockets sécurisées (SSL/TLS) (1/2)

- Plusieurs spécifications
  - *Secure Socket Layer* (SSL) et *Transport Layer Security* (TLS - variante de SSLv3 normalisée par l'IETF)
  - Les principes de base sont les mêmes pour SSL et TLS
- Motivation
  - Garantir la confidentialité et l'intégrité des communications sur un canal TCP, permettre l'authentification du client et/ou du serveur
  - Exemples : envoi d'un numéro de carte de crédit sur un serveur Web de commerce électronique, consultation d'un compte bancaire (HTTPS), connexion par identifiant/mot de passe à un service (serveur SMTP, serveur POP/IMAP, …)
- Implémentation sous forme de bibliothèque
  - Au dessus de l'interface socket de la couche transport

Sockets sécurisées (SSL/TLS) (2/2)

- Etablissement du dialogue
  - Etablissement de la connexion TCP puis …
  - Le client envoie un message « SSL Hello », le serveur répond en envoyant un certificat qui contient sa clé publique (Spub)
  - Le client peut ainsi vérifier l'authenticité (de la clé) du serveur et génère une clé secrète de session (MS), il la chiffre avec Spub et envoie le résultat (EMS) au serveur
  - Le serveur obtient la clé MS en déchiffrant EMS avec Spriv
- Transmission des données
  - Les flots d'octets (applicatifs) envoyés dans chaque sens sont découpés en enregistrements (*records*) de taille bornée
  - Les enregistrements sont chiffrés (et déchiffrés) avec la clé MS
- En pratique, le protocole est plus complexe
  - Négociation des techniques de cryptographie au début du dialogue
  - Des précautions complémentaires (plusieurs clés, codes d'authentification des messages …) sont nécessaires pour déjouer des attaques élaborées

Sécurité des réseaux
SSH (Secure Shell)

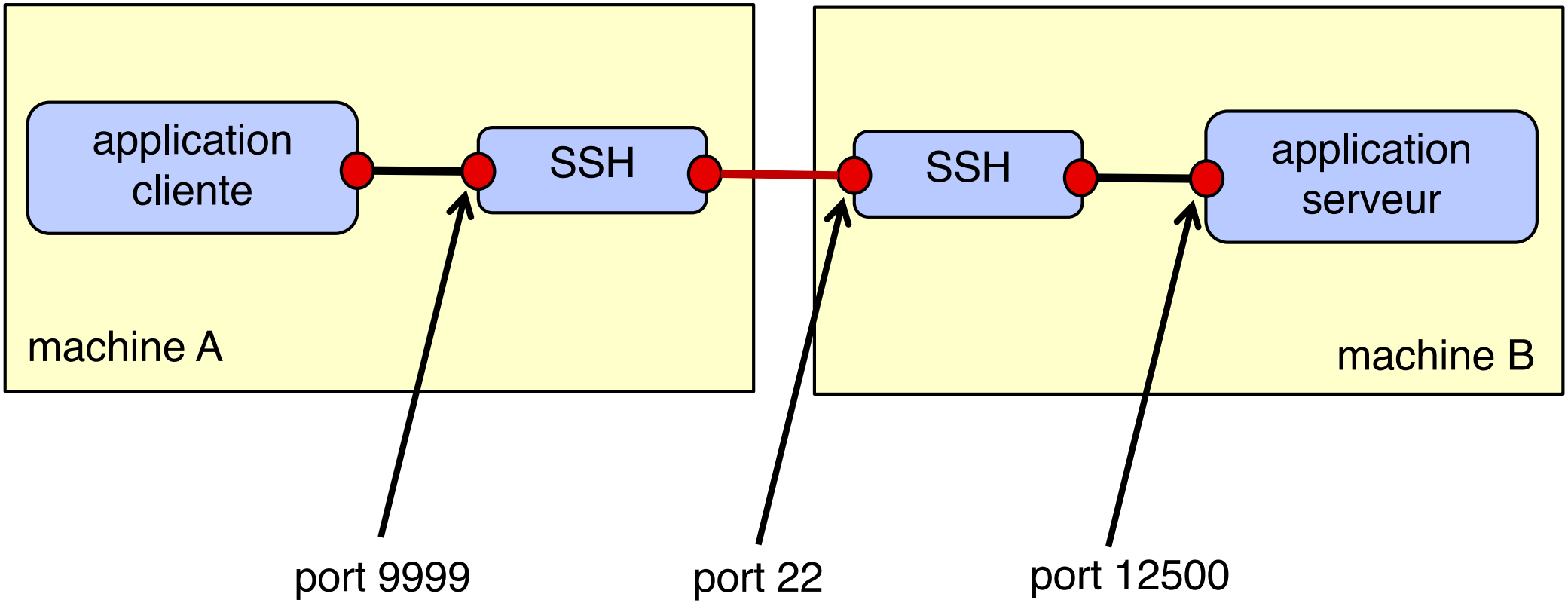
- Motivation
  - Applications : login à distance, exécution de commandes/applications graphiques à distance, transfert de données
  - Garanties : confidentialité et intégrité des informations échangées
- SSH v2 englobe un empilement de trois protocoles
  - SSH-TRANS : canal de communication chiffré
    - Basé sur TCP, avec vérification de l'intégrité des messages
  - SSH-AUTH : authentification d'un utilisateur
    - Par mot de passe, clé publique, ou éventuellement liste de couples <utilisateur, machine> fiables
  - SSH-CONN : exécution/interaction à distance, notion de tunnel SSH (et redirection de ports) pour encapsuler des protocoles non sécurisés
  - Références : voir les RFC n° 4251 à 4254
- Applications basées sur SSH
  - Exemples : SFTP et SCP pour les transferts sécurisés de fichiers

Tunnel (ou redirection) SSH (1/3)

- Exemple
  - On veut sécuriser les échanges entre une application cliente sur une machine A et une application serveur sur une machine B (sans modifier le code source des programmes)
  - On suppose que l'utilisateur de la machine A (cliente) dispose d'un compte (accès SSH) sur la machine B (serveur) ou, au minimum sur une machine C appartenant au même réseau que le serveur (en supposant que les communications entre B et C sont sécurisées).
  - Normalement (sans sécurisation), le client contacte le serveur sur le port TCP N° X (par exemple 12500)
- Etapes pour la mise en place d'un tunnel SSH
  - Choix d'un port libre (Y) sur le client (par exemple Y=9999)
  - Création du tunnel :

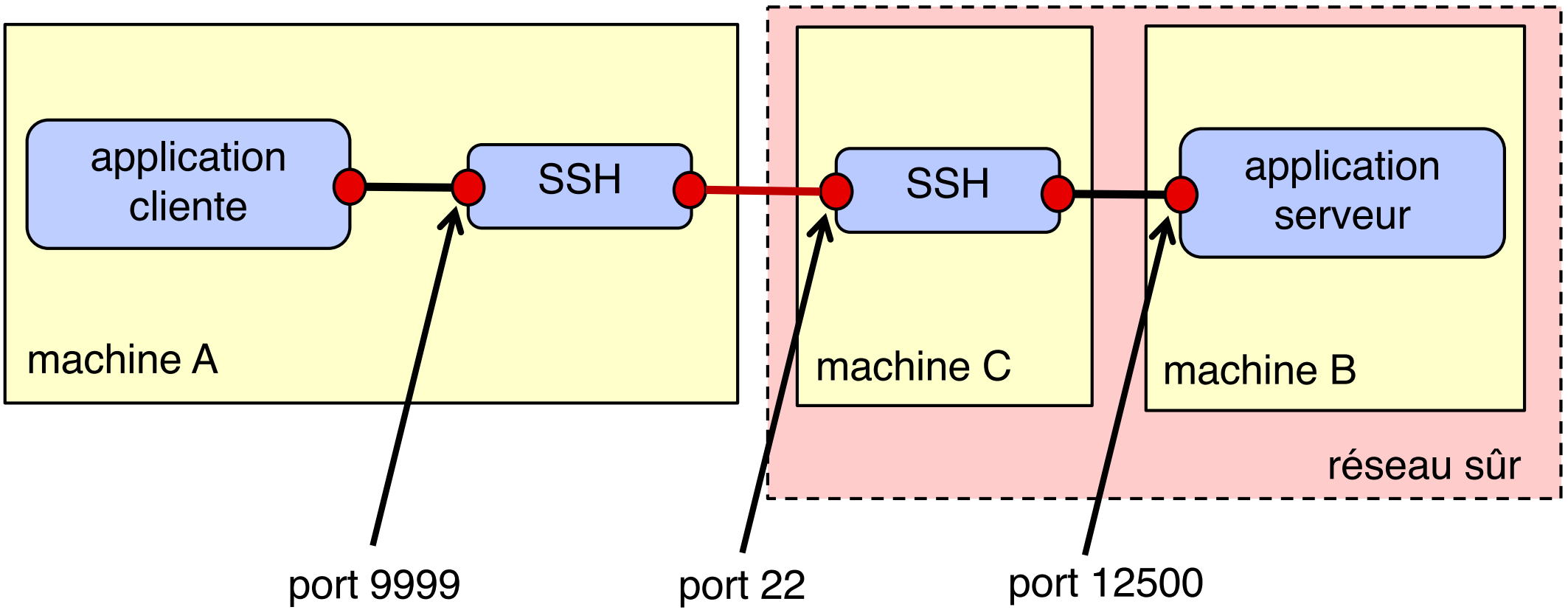
    ssh –L Y:<machine serveur>:X <machine avec compte SSH>
  - Configurer l'application cliente pour qu'elle contacte le serveur à l'adresse localhost:9999 plutôt qu'à l'adresse <machineB>:12500

Tunnel (ou redirection) SSH (2/3)



ssh –L 9999:<machine B>:12500 <login sur machine B>@<machine B>

Tunnel (ou redirection) SSH (3/3)



ssh –L 9999:<machine B>:12500 <login sur machine C>@<machine C>

Tunnel SSH + SOCKS (1/2)

- Motivation : restrictions d'un tunnel SSH simple
  - Nécessité de créer un tunnel par port redirigé
  - Nécessité de modifier la configuration de chaque application
  - ... (fausses alertes d'authentification liées aux certificats cryptographiques)
- SOCKS
  - Essentiellement un protocole applicatif permettant d'interagir avec un proxy pour créer des connexions
  - Un proxy SOCKS écoute par défaut sur le port TCP n° 1080
- Principe de SSH + SOCKS
  - Création d'un proxy sur la machine locale
  - Lorsqu'une application a besoin d'une nouvelle connexion, elle passe par le proxy pour en demander la création
  - Comment configurer l'application pour interagir avec le proxy ?
    - Explicitement, si elle gère le protocole SOCKS
    - Ou de façon transparente via une bibliothèque dynamique qui intercepte (et modifie) les appels système liés aux communications réseau
  - Une fois la connexion mise en place, le fonctionnement du proxy est transparent pour les deux extrémités du dialogue applicatif

Tunnel SSH + SOCKS (2/2)

- Exemple
  - On reprend l'exemple précédent avec le tunnel SSH
    - L'utilisateur U d'une machine cliente A veut établir une liaison sécurisée vers une machine B (sur laquelle U dispose d'un accès SSH)
    - Et le serveur SSH exécuté sur B servira de relais pour dialoguer avec un autre service (hébergé sur B ou sur une autre machine C)
  - Lancement du proxy SSH+SOCKS sur la machine A (port 1080) :
    - ssh –D 1080 <login sur la machine B>@<nom de la machine B>
  - Lancement et configuration de l'application cliente :
    - Soit explicitement : spécifier le proxy SOCKS localhost sur le port 1080
    - Soit implicitement en lançant l'application via un programme assistant permettant d'intercepter et de modifier les communications réseau, tel que tsocks : `tsocks monapplication`

Compléments sur les tunnels

- Avertissement
  - Par défaut, le trafic DNS n'est pas forcément redirigé dans le tunnel
  - Les requêtes/réponses DNS passent donc en clair sur le réseau local et peuvent être observées, voire modifiées
  - La configuration SOCKS de certaines applications (par exemple Firefox) permet de rediriger les requêtes DNS dans le tunnel

- Autres utilisations et bénéfices des tunnels
  - Contourner un pare-feu (firewall) très restrictif
  - Utiliser une machine qui héberge un serveur SSH comme proxy (quelle que soit l'application utilisée) afin de :
    - ne pas exposer son identité (adresse IP de la machine cliente) au service contacté
    - accéder à un service réservé à certaines adresses IP/localités
    - …

Un autre exemple pour la sécurité des communications : Sécurisation au niveau de la couche 3 (réseau)

- Caractéristiques:
  - Transparence par rapport aux divers protocoles des couches *transport* et *application*
  - La sécurisation n'est pas forcément appliquée d'un bout à l'autre du chemin
  - Possibilité de sécuriser/rediriger l'ensemble du trafic réseau d'une machine
  - Application : Réseau privé virtuel (*Virtual Private Network* ou *VPN*)

- Exemple : protocole IPsec
  - Nécessite l'établissement d'une « connexion logique » entre le point de chiffrement et le point de déchiffrement
  - Contenu d'un paquet (mode tunnel, ESP) :
    - En-tête IP (classique)
    - En-tête IPsec (permet d'identifier la connexion et la clé de déchiffrement associée, ainsi que le numéro de séquence du paquet)
    - Contenu chiffré : essentiellement paquet IP (complet) à protéger + bourrage
    - Code (signature) pour l'authentification et l'intégrité du message

Détails sur IPsec

- Deux modes d'utilisation : transport et tunnel

- Mode transport
  - Seul le contenu d'un paquet IP est sécurisé (données des couches transport et application)
  - L'en-tête IPSEC se trouve derrière l'en-tête du paquet IP

- Mode tunnel
  - L'ensemble du paquet IP est sécurisé
  - Le paquet IP de départ est complètement encapsulé au sein d'un paquet IPsec
  - Plus complexe mais plus général
    - Permet par exemple des communications sécurisées sans que le destinataire final utilise lui même IPsec (encapsulation/désencapsulation au niveau de routeurs)
  - Mode le plus utilisé, notamment pour les VPN

Détails sur IPsec (suite)

- IPsec regroupe un ensemble de protocoles
- Deux principaux protocoles de sécurité : AH et ESP

- AH (*Authentication Header*)
  - Garanties d'authentification de l'émetteur
  - Garanties d'intégrité
  - Pas de garanties de confidentialité

- ESP (*Encapsulating Security Payload*)
  - Garanties d'authentification de l'émetteur
  - Garanties d'intégrité
  - Garanties de confidentialité

Détails sur IPsec (suite)

- Exemple n° 1 : mode transport avec AH (IPv4)

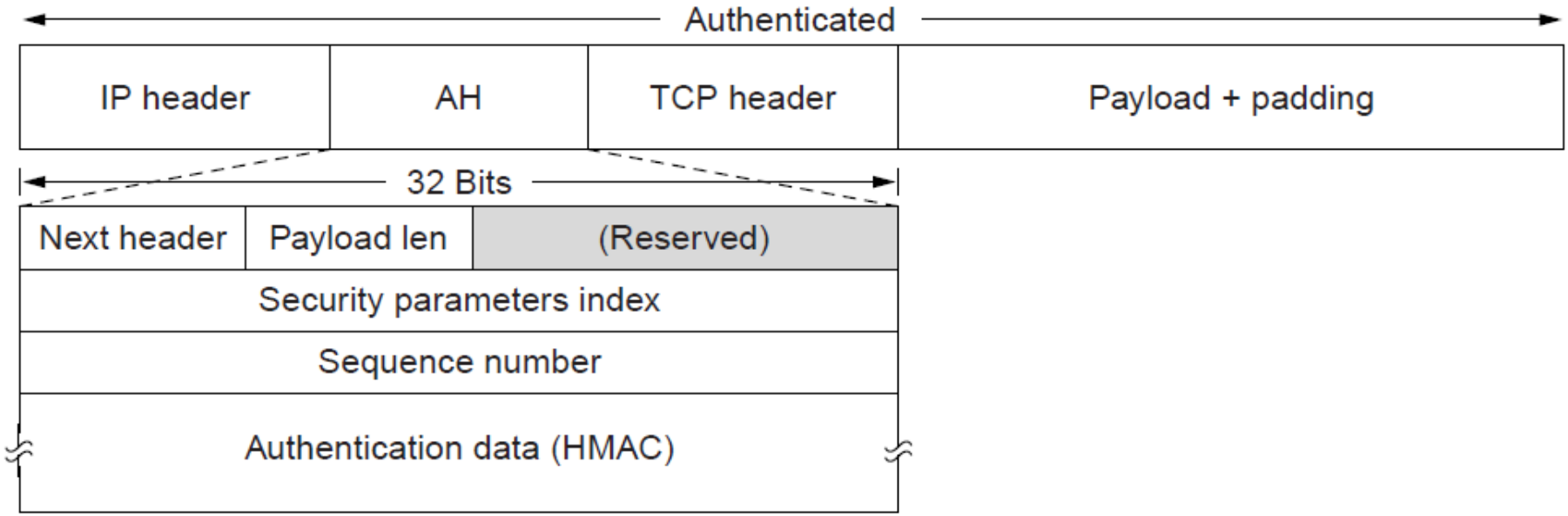
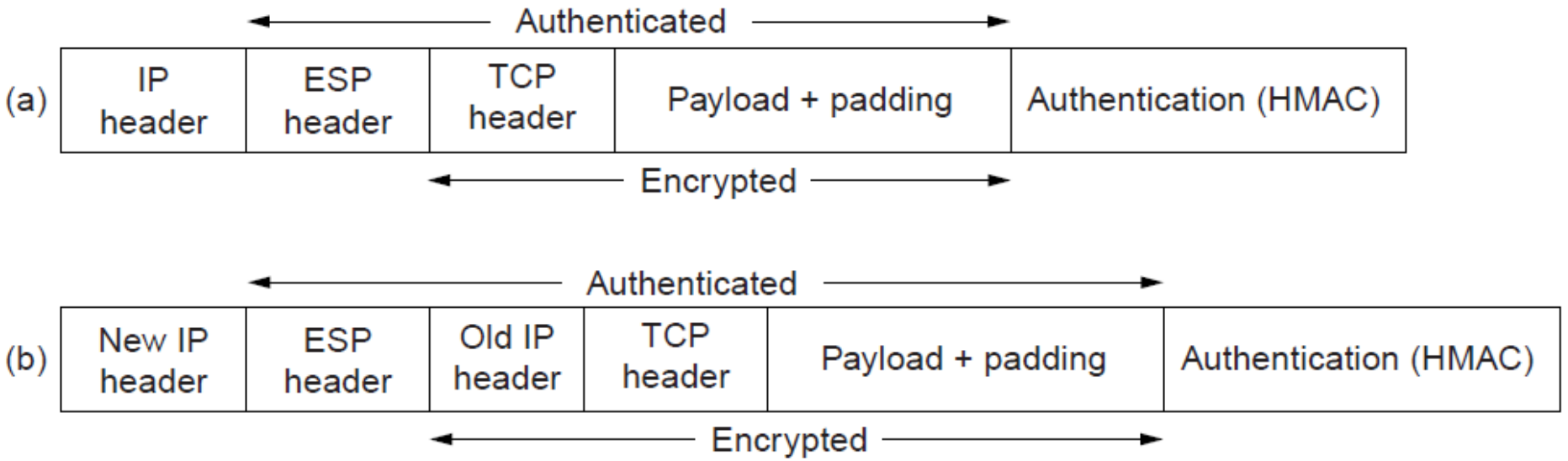


Schéma extrait de :
Andrew Tanenbaum and David Wetherall, *Computer Networks*, (5th edition), Prentice Hall, 2011

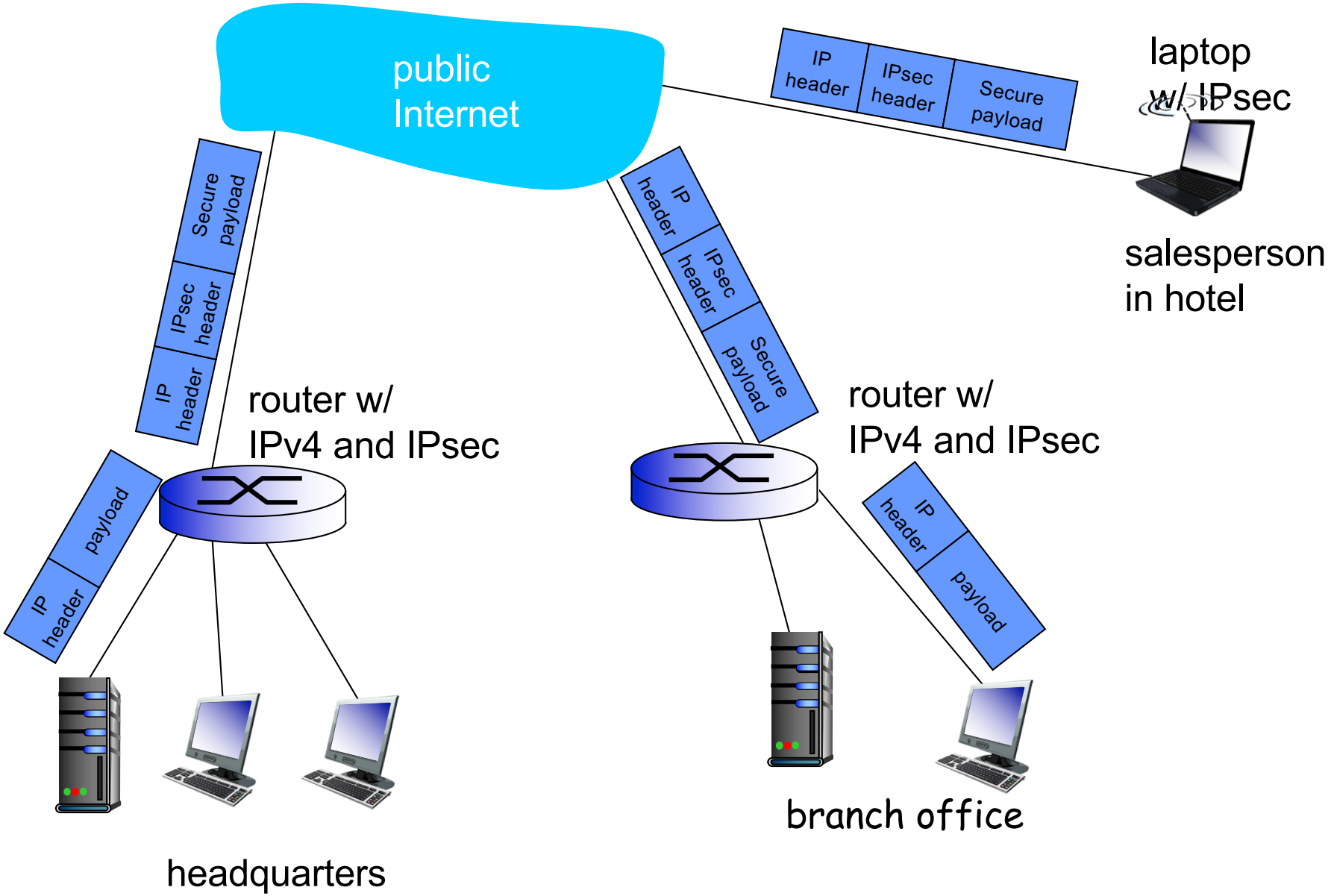
Détails sur IPsec (suite)

- Exemple n° 2 :
  - (a) mode transport avec ESP
  - (b) mode tunnel avec ESP (combinaison la plus courante pour IPsec)



Schémas extraits de :
Andrew Tanenbaum and David Wetherall, *Computer Networks*, (5th edition), Prentice Hall, 2011

VPN


public
Internet
IP header
IPsec header
Secure payload
laptop
w/ IPsec
salesperson
in hotel
router w/
IPv4 and IPsec
router w/
IPv4 and IPsec
IP header
payload
branch office
headquarters