Cours de Réseaux

7 – La couche application

Rappel : le modèle hybride

5	Couche application
4	Couche transport
3	Couche réseau
2	Couche liaison de données
	(Sous-couche contrôle d'accès au canal)
1	Couche physique

Rappel - la couche applications

- Réalisation de protocoles « utilisateur » comme par exemple dans l'internet :
  - HTTP (pages web)
  - SMTP (mail électronique)
  - FTP (transfert de fichiers)
  - NNTP (Network News)
- Réalisation de protocoles destinés à d'autres services réseau, par exemple
  - RCP (*Remote Procedure Call*)
  - DNS (traduction de noms vers des adresses IP)

DNS : Domain Name Service

- Traduction des *noms* (par ex : `www.ens-cachan.fr`) vers des adresses IP.
- Pourquoi utiliser des noms et pas des adresses IP ?
  - plus facile à mémoriser
  - possibilité de changer la machine (et le numéro IP) associé à un nom (par exemple pour un serveur web)
  - possibilité d'avoir plusieurs machines identiques associées au même nom (ferme de serveurs, par exemple Google)
  - les machines d'un seul domaine « logique » peuvent être distribuées dans le monde (par exemple : le projet Debian)

Les noms DNS

Noms hiérarchiques, partie la plus spécifique à gauche. Exemple :

`www.dptinfo.ens-cachan.fr`

La partie la plus générale (ici : `fr`) est le *top level domain* - TDL (domaine de premier niveau).

La liste des TDL est gérée par la *Internet Corporation for Assigned Names and Numbers* - ICANN (organisme sous contrôle du ministère américain d'économie).

Les domaines de premier niveau

– Les domaines génériques classiques :

com sites commerciaux

edu enseignements

mil armée américaine

gov gouvernement américain

org organismes à but non lucratif

net gestion du réseau

– TDL génériques récemment crées : `info`, `biz`, `museum`, `aero`, ...

– TDL par pays : `us` (peu utilisé), `fr`, `de`, ...

Gestion des noms

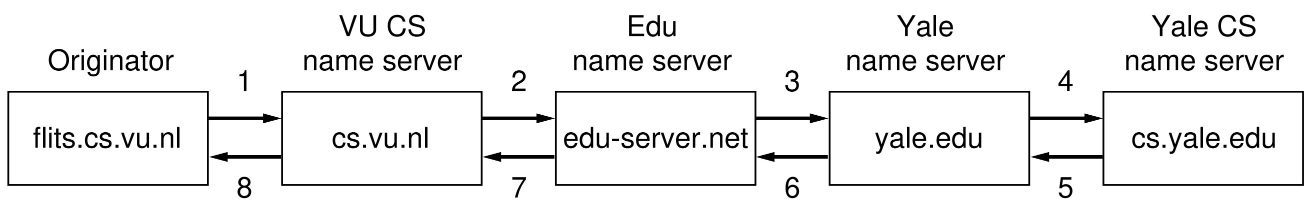
- La gestion des domaines est sous-contractée à des revendeurs de noms de domaine.
- La plupart des TDL ont un règlement interne (par exemple `org`), d'autres sont de libre acces (par exemple `name`).
- *Cybersquatting* : Enregistrer des noms internet qui ressemblent à des noms de marques, avec le but de les revendre.
- Les pays gèrent leur domaine de façon différente. Par exemple, certains pays (UK, Japon) ont des sous-domaines `.co` (commercial) et `.ac` (académique).

Les serveurs de noms

- Normalement, un machine sur internet utilise un serveur de noms (UNIX : adresse du serveur spécifiée dans `/etc/resolv.conf`).
- Un serveur de noms connaît normalement que les noms de son domaine.
- Résolution de noms récursive (le cas normal) : Si un serveur ne connaît pas un nom il demande au serveur du TDL, qui fait passer la demande.

Recherche de noms

Exemple : chercher l'adresse de `cs.yale.edu`



Serveurs DNS : Caching

– Les serveurs DNS mémorisent les entrées reçues pour un certain temps.
– Le temps de vie maximal d'une copie est indiquée dans l'entrée (par exemple 1 journée pour des données qui varient très peu).
– Une réponse qui vient d'une entrée mémorisée par un serveur est *non officielle*.
– Seul le serveur DNS responsable pour un domaine peut donner des réponses officielle pour ce domaine.

DNS : Réponses officielles et non officielles

```
% nslookup
> figue.lsv.ens-cachan.fr
Server:         212.27.54.252
Non-authoritative answer:
Name:   figue.lsv.ens-cachan.fr
Address: 138.231.81.83
> server ariane.ens-cachan.fr
Default server: ariane.ens-cachan.fr
> figue.lsv.ens-cachan.fr
Server:         ariane.ens-cachan.fr
Name:   figue.lsv.ens-cachan.fr
Address: 138.231.81.83
```

DNS Spoofing

- Méthode pour dérouter des connexions internet (attaque de l'homme au milieu). Exemple : A veut parler avec le serveur B.
- L'attaquer donne à un serveur de noms une adresse IP truquée pour le serveur B.
- L'attaquer n'a pas besoin d'attendre le moment où A fait la demande car il peut exploiter le fait que le serveur mémorise les réponses.
- Une solution : DNS sécurisé (authentification par clefs cryptographiques)
- Conséquence : réaliser l'authentification au niveau application quand nécessaire (exemple : SSL, utilisé par `https`).

DNS : requête serveur mail

```
% nslookup
> set type=MX
> ens-cachan.fr
Server:         212.27.54.252
Address:        212.27.54.252#53

Non-authoritative answer:
ens-cachan.fr   mail exchanger = 10 ariane.ens-cachan.fr.
ens-cachan.fr   mail exchanger = 10 ariane2.ens-cachan.fr.
```

DNS : requête serveur DNS

```
% nslookup
> set type=NS
> ens-cachan.fr
Server:         212.27.54.252
Address:        212.27.54.252#53

Non-authoritative answer:
ens-cachan.fr   nameserver = ariane.ens-cachan.fr.
ens-cachan.fr   nameserver = rubis.cri.ens-cachan.fr.
ens-cachan.fr   nameserver = dufy.aquarel.fr.
ens-cachan.fr   nameserver = ns2.nic.fr.
```

Les services Internet

Les services suivent chacuns son *protocole d'application* : HTTP, SMTP, FTP, ...

Par convention, *si* on veut fournir un certain service standard on l'attache au numéro de port selon `/etc/services`.

Le service peut être réalisé par un processus UNIX dédié, ou par le *superserveur* `inetd` (voir le fichier `/etc/inetd.conf`).

Lancement des services au démarrage de la machine.
(Linux : voir les scripts dans les répertoires `/etc/rc?.d`.)

Email

Déroulement de l'envoi d'un message

- Composition du message sur un *client* (mutt, pine, kmail, netscape, evolution, ...).
- Le client passe le message au *Mail Transport Agent* (MTA) qui tourne (normalement) sur sa machine locale.
- Le MTA modifie éventuellement l'entête (par exemple normalisation de l'adresse de l'émetteur).
- Le MTA obtient l'adresse de la machine récepteur correspondant à l'adresse du destinataire par une requête DNS.

Déroulement de l'envoi d'un message (2)

– Le MTA ouvre une connexion TCP avec le MTA récepteur (port 25), et lui transmet le message par le protocole d'application SMTP (*Simple Mail Transfer Protocol*). Éventuellement file d'attente pour l'envoi.
– Le MTA récepteur peut modifier l'entête (réécriture de l'adresse du destinataire, ajout d'une ligne informative sur la réception, ...).
– Dans le cas de livraison locale : le MTA écrit le message dans un fichier qui peut être lu par l'utilisateur destinataire (`mbox`) dans un répertoire système (`/var/spool/mail/`).

Entêtes mail (1)

Champs concernant le transport :

Header	Meaning
To:	E-mail address(es) of primary recipient(s)
Cc:	E-mail address(es) of secondary recipient(s)
Bcc:	E-mail address(es) for blind carbon copies
From:	Person or people who created the message
Sender:	E-mail address of the actual sender
Received:	Line added by each transfer agent along the route
Return-Path:	Can be used to identify a path back to the sender

Entêtes mail (2)

Quelques d'autres champs :

Header	Meaning
Date:	The date and time the message was sent
Reply-To:	E-mail address to which replies should be sent
Message-Id:	Unique number for referencing this message later
In-Reply-To:	Message-Id of the message to which this is a reply
References:	Other relevant Message-Ids
Keywords:	User-chosen keywords
Subject:	Short summary of the message for the one-line display

Entêtes mail (3)

MIME (Multipurpose Internet Mail Extension)

Permet l'envoi de messages écrits dans dans des alphabets d'autres que ASCII, des documents dans des formats autres que du texte.

Header	Meaning
MIME-Version:	Identifies the MIME version
Content-Description:	Human-readable string telling what is in the message
Content-Id:	Unique identifier
Content-Transfer-Encoding:	How the body is wrapped for transmission
Content-Type:	Type and format of the content

Exemple d'une session SMTP (1)

```
% telnet smtp.free.fr smtp
Trying 212.27.48.4...
Connected to smtp.free.fr.
Escape character is '^]'.
220 smtp8-g19.free.fr ESMTP Postfix
HELO seneca
250 smtp8-g19.free.fr
MAIL FROM: <treinen@free.fr>
250 Ok
RCPT TO: <treinen@free.fr>
250 Ok
```

Exemple d'une session SMTP (2)

```
DATA
354 End data with <CR><LF>.<CR><LF>
From: treinen@free.fr
To: treinen@free.fr

Ceci est un test.
.
250 Ok: queued as DA1E0555F
QUIT
221 Bye
Connection closed by foreign host.
```

Traitement des messages par le MTA récepteur

- Expansion des listes de diffusion, par exemple `ens-info06@lists.crans.org`
- Récrire des alias des noms de récepteurs, par exemple `Ralf.Treinen@...` ⇒ `treinen@...`
- Renvoie vers des autres MTA (voir plus tard).
- Filtrage du contenu (filtres spam, ...)
- Filtrer par un filtre défini par l'utilisateur destinataire (programme `vacation`, `procmailrc`)

Filtrage par `procmailrc`

Liste de règles à appliquer sur les messages reçus.

Sélection par expression régulières appliqués à certains champs de l'entête, ou au corps du message.

Actions possibles : classer le mail dans un fichier, passer le message à un autre programme.

Transfert des messages en plusieurs étapes

- Les grands organismes on souvent un service de mail central, qui renvoie les message reçus à des MTA des départements. Permet d'avoir de adresses du style `Bill.Gates@microsoft.com`
- Les MTA des départements sont souvent derrières des firewalls et ne peuvent pas être contactés directement par un émetteur. Par exemple, le MTA du LSV est derrière un firewall, la réception des mails au LSV passe par le MTA de l'école.
- Centralisation du traitement de mail sur une machine (par exemple un filtre Spam).

Transfert des messages en plusieurs étapes (2)

- Simplification des MTA sur les machines utilisateurs (envoyer tous les messages qui ne sont à délivrer localement vers un *smarthost*)
- Connexion intermittente des machines utilisateurs (portable, connexion modem) : Réception des messages sur un serveur, le client récupère les message avec un protocole auxiliaire (POP, IMAP).

  Dans le cas du programme `fetchmail` : récupérer les message d'un serveur (souvent POP), les passer au MTA de la machine locale.

Transfert des messages en plusieurs étapes (3)

- Rédirection des messages comme spécifié par un utilisateur dans son fichier `.forward`
- Rédirection vers des organismes tiers. Par exemple : `treinen@acm.org` ⇒`treinen@lsv.ens-cachan.fr`
- Gestion des listes de diffusion

Extrait d'entête d'un mail reçu

```
Delivered-To: treinen@lsv.ens-cachan.fr
Received: from ariane.ens-cachan.fr (ariane.ens-cachan.fr [138.231.176.4])
        by olive.lsv.ens-cachan.fr (Postfix) with ESMTP id 5082778375
        for <treinen@lsv.ens-cachan.fr>; Wed, 10 Jan 2007 08:04:23 +0100 (CET)
Received: from mx1.aist.go.jp (mx1.aist.go.jp [150.29.246.133])
        by ariane.ens-cachan.fr (Postfix) with ESMTP id E914241FC
        for <treinen@lsv.ens-cachan.fr>; Wed, 10 Jan 2007 08:04:20 +0100 (CET)
Received: from rqsmtp1.aist.go.jp (rqsmtp1.aist.go.jp [150.29.254.115])
        by mx1.aist.go.jp  with ESMTP id l0A72lUM005509
        for <treinen@lsv.ens-cachan.fr>; Wed, 10 Jan 2007 16:02:47 +0900 (JST)
        env-from (ohsaki@ni.aist.go.jp)
From: Hitoshi Ohsaki <ohsaki@ni.aist.go.jp>
```

Problèmes d'abus : le *Spam*

Spam = messages non-sollicités et envoyés en très grand nombre

Nombre estimé de messages spam envoyés, décembre 2006 : 85 milliards par jour (il y en a beaucoup qui ne sont pas délivrés).

Constitution des listes d'adresses par les spammeurs :
- Sites web (spiders : logiciel qui traverse le web en suivant les liens)
- Archives web des listes de discussions (news)
- Deviner les adresses, en utilisant les noms des domaines et des noms communs d'utilisateurs

- Virus qui récupèrent des listes d'adresses des annuaires utilisateurs
- Astuces pour obtenir la confirmation de la validité d'une adresse web (par exemple : demander un message de désabonnement d'une liste de diffusion fictive).

Techniques d'envoi de spam

- Utiliser des comptes créés pour l'occasion
- Fausser des entêtes
- Utiliser des *relais ouverts* : MTA mal configurés qui acceptent des messages des utilisateurs sans vérification de l'origine.
- *Spambots* : réseaux de machines infiltrés (virus) utilisées pour l'envoi des messages, souvent commandés par des serveurs IRC.

Techniques de défense contre le spam

Tout fournisseur d'accès internet doit interdire l'utilisation des son service pour l'envoi du spam.

Éviter la création des comptes sur des services email (hotmail, yahoo, etc.) par des robots : afficher un mot de façon tordue (ou l'annoncer par audio), et demander à l'utilisateur d'entrer le mot.

Contre-mesure : produire une page web et faire des utilisateurs décoder le mot, en promettant un service gratuit sur inscription.

Technique de défenses contre le spam

- Ne pas accepter des messages qui viennent des spammeurs connus (*blacklists* maintenus par des organismes, comme Spamhaus)

  Techniques pour établir des listes de spammeurs ou de messages spam :
  - Rapport par des utilisateurs
  - *Honeypots* : systèmes qui se présentent comme des relais ouverts mais qui ne servent qu'à obtenir des informations sur des malfaiteurs
  - *Spamtraps* : adresses qui ne servent qu'à attirer des spams, utilisés pour établir des collections de messages spam.

Défense contre le spam (2)

- *Greylisting* : quand une machine inconnue se connecte pour la première fois à un port SMTP la connexion est refusée. Normalement, un MTA doit faire un nouvel essai plus tard, mais les spammeurs ne le font souvent pas (par manque de ressources).
- *SMTP Callback Verification* : Le MTA fait de sa part ne connexion au MTA de l'émetteur prétendu et vérifie qu'il existe, et accepte des mails. Controversé.

Défense contre le spam (3)

– Filtrer les messages selon les entêtes : par exemple, l'argument de HELO doit être un FQDN (fully qualifie domain name), ou une adresse IP entre parenthèses :

Valide :	HELO [127.0.0.1]
Invalide :	HELO localhost
Invalide :	HELO 127.0.0.1
Invalide :	HELO domain.tld
Invalide :	HELO friend
Invalide :	HELO -232975332

Défense contre le spam (4)

- Filtrer les messages selon le contenu : analyse statistique des mots qui paraissent dans le message (outil `bogofilter`).

  Contre-mesures :
  - utiliser des images
  - obfuscer le texte en utilisant des balises HTML
  - remplaçer par exemple `Viagra` par `\/iagra` ou `V1agra`.
  - Utiliser des phrases récupérées de la page web de la victime.

Exemple d'un filtrage par `spamassasin`

```
Content preview:  Are you an Industry Expert? (accounting, computers,
  media, technology...) Join our Network of Industry Experts and be paid

Content analysis details:   (15.4 points, 5.0 required)

 pts description
---- --------------------------------------------------
 2.2 Invalid Date: header (not RFC 2822)
 1.2 Date: is 12 to 24 hours before Received: date
 0.0 BODY: HTML included in message
 0.0 BODY: Message only has text/html MIME parts
 0.2 BODY: HTML title contains no text
-1.1 BODY: Bayesian spam probability is 1 to 5%
```

```
          [score: 0.0353]
0.2 RAW: base64 attachment does not have a file name
1.9 RAW: Message text disguised using base64 encoding
0.2 RAW: Extra blank lines in base64 encoding
0.2 RBL: Envelope sender in abuse.rfc-ignorant.org
2.6 RBL: Envelope sender in dsn.rfc-ignorant.org
2.0 RBL: SORBS: sent directly from dynamic IP address
         [87.17.166.4 listed in dnsbl.sorbs.net]
1.6 Received via a relay in bl.spamcop.net
     [Blocked - see <http://www.spamcop.net/bl.shtml?87.17.166.4>]
1.7 RBL: Envelope sender in postmaster.rfc-ignorant.org
1.9 RBL: NJABL: dialup sender did non-local SMTP
     [87.17.166.4 listed in combined.njabl.org]
0.4 Subject: base64 encoded encoded unnecessarily
```

Problèmes de sécurité : Signatures et Chiffrement

PGP : PrettY Good Privacy, Phil Zimmerman, 1991
GPG : GNU Privacy Guard

- Chiffrement symétrique : la même clef utilsée pour chiffrement et déchiffrement. Exemples : DES, AES. Très rapide.
- Chiffrement asymétrique : deux clefs, une pour chiffrement, une pour le déchiffrement. Exemples : RSA, DSA. Cher en calcul.

Utilisation du chiffrement asymétrique (1)

– Assurer le secret : Bob publie la partie publique de sa clef (page web, serveur de clefs). Quand Alice veut lui envoyer un message secret elle récupère la clef publique de Bob et l'utilise pour chiffrer le message. Seul Bob connaît la clef de déchiffrement correspondante.
  En pratique : Utiliser la clef asymétrique seulement pour transmettre une clef symétrique, puis chiffrer le message avec la clef symétrique.

Utilisation du chiffrement asymétrique (2)

– Signer des messages : Avant envoyer un message, Bob le chiffre avec sa clef privée. Tout le monde peut récupérer la clef publique correspondante et déchiffrer le message.
  En pratique : Calculer un *hash* du message, ne chiffrer que la valeur de hachage.
  Attention à la méthode de hachage utilisée (MD4 non, MD5 plutôt non, SHA-1 et SHA-2 oui)

Certification de clefs

Comment être sûr de la clefs d'une autre personne ?

- Utiliser des canaux fiables : téléphoner si on connaît la personne et demander l'empreinte de la clef, imprimer l' empreinte de la clef sur sa carte de visite.
- Les clefs peuvent porter elles-mêmes des signatures :
  - « Authorités de certification »
  - Établir un « web of trust » : Signer les clefs des personnes lors d'un rencontre (*key signing party*), vérifier une clef en établissant un chemin dans le graphe des signatures.

Utilisation de gpg

- Générer des paires de clefs
- Gérer sa « trousse de clefs » (keyring)
- Obtenir des clefs d'un serveur, envoyer vers un serveur
- Signer des clefs
- Chiffrer, déchiffrer des fichiers (fonctionnalité intégrée dans tout client de mail sérieux)

Pour en savoir plus : Voir le *GNU Privacy Handbook*.

Protocoles de connexion à distance : telnet

Protocoles d'accès à un interpréteur de commande (*shell*) sur une machine.

telnet

Existe en UNIX depuis 1969

Pas à confondre avec l'outil *telnet* qui permet d'exécuter « à la main » un protocole d'application basé texte.

telnet (2)

Très insécure car :
- Pas de chiffrement des communications : possibilité d'espionner des donnés (mot de passe !) avec un « sniffeur »
- Pas d'authentification (des machines) : possibilité de récuperer des données par une attaque de l'homme au milieu
- Dans les implémenations souvent des vulnérabilités (dépassement de la pile, etc.)

N'est pratiquement plus utilisé aujourd'hui.

Protocoles de connexion à distance : rlogin

- Introduit par BSD Unix.
- Restreint à l'utilisation entre machines UNIX (contrairement à telnet)
- **`rcp`** : Copier des fichiers entre machines
- **`rsh`** : Exécuter une commande sur une autre machine

Exemples :

```
rsh serveur.lsv.ens-cachan.fr lpr < fichier
rcp fichier1 serveur.lsv.ens-cachan.fr:fichier2
```

rlogin (2)

Permet la même utilisation que telnet, mais aussi accès sans mot de passe :

- Fichier `~/.rlogin` : Un utilisateur autorise les connexions d'un certain utilisateur à partir d'une certaine machine.
- Fichier `/etc/hosts.equiv` : Donne des noms des machines à partir desquelles un utilisateur avec le même nom d'utilisateur peut se connecter.

rlogin (3)

Problèmes :

- Les mêmes que *telnet* (fait confiance au réseau)
- De plus : fait confiance à l'authentification des utilisateurs sur la machine client.

N'est pratiquement plus utilisé aujourd'hui.

ssh : secure shell

- Existe en versions propriétaires et libres (OpenSSH)
- Disponible sur des UNIX (dont Linux), Mac OS, Windows, ...
- Fonctionnalité de rlogin : connexion de shell, exécution des commandes à distance, copier des fichiers entre machines
- *Port forwarding*, ou *tunneling* : permet à un utilisateur de transférer un autre protocole par un *tunnel* ssh, utilisé par exemple pour accès à distance à un un service qui normalement n'est que disponible pour accès local (serveur mail, serveur web).
  Aussi utilisé par X (graphisme) pour l'affichage des applications graphiques à distance
- Utilisé par des autres services réseau comme rsync, CVS, ...

ssh : secure shell (2)

- Authentification des machines par chiffrement asymétrique
- Utilise chiffrement asymétrique pour échanger une clef de session : clef symétrique utilisé pour chiffrer la communication
- Authentification des utilisateurs par clefs asymétriques (méthode préférée), ou par mot de passe (transfert chiffré).

ssh : secure shell (3)

Répertoire `/etc/ssh` contient

– fichier avec des options de configuration
  Par exemple, l'administrateur du système peut restreindre des méthodes d'authentification permises.
– fichier avec des clefs publiques des machines connues à l'administrateur.
– fichier avec la clef publique de la machine (à utiliser pour authentifier cette machine lors d'une connexion à distance).

ssh : secure shell (4)

Répertoire `.ssh` d'un utilisateur contient

- `config` : options de configuration
- un fichier contenant les clefs privées de l'utilisateur, normalement protégée par une *passphrase*
- `authorized_keys` : les clefs publiques pour lesquelles on accepte des connections
- `known_hosts` : les clefs publiques des machines utilisées pour l'authentification des machines