3. Courrier électronique

M1. Outils de l'Internet
lundi 4 octobre 2010

victor.poupet@lif.univ-mrs.fr

Présentation

Histoire

- Plus vieux qu'Internet (et même qu'ARPANet)
- 1961 : utilisation d'une machine par plusieurs utilisateurs distants (OS : CTSS) → e-mail en 1965
- 1966 : e-mail à travers un résau (machines similaires)
- 1969 : ARPANet favorise le développement de l'e-mail

Le mail devient vite l'utilisation principale d'ARPANet

Format des messages

- RFC 5322, 2045-2049
  - Multipurpose Internet Mail Extensions (MIME)
  - Un en-tête (*header*)
  - Un corps (*body*)

En-tête

- Différents champs (`nom: valeur`)
- Uniquement des caractères ASCII 7 bits (encodage MIME pour le reste)
- RFC 5322

Champs obligatoires

- `From:` adresse [et nom] de l'auteur
- `To:` adresse(s) [et nom(s)] du ou des destinataires
- `Subject:` description du message
- `Date:` date et heure de l'envoi du message
- `Message-ID:` identifiant unique du message

Remarque : les champs `To` et `From` ne correspondent pas nécessairement aux adresses des véritables sources et cibles...

D'autres champs...

- `CC:` copie carbone
- `BCC:` copie carbone cachée (*blind*)
- `Received:` permet de retracer le trajet d'un message
- `Content-Type:` un type MIME
- `Reply-To:` adresse de réponse
- `In-Reply-To`, `References`, ...

246 champs définis par l'IANA...

Corps

- Historiquement : ASCII 7 bits
- Aujourd'hui ASCII 8 bits est couramment supporté
- 2 encodages possibles :
  - *quoted printable*
  - *base64*

Encodages : 7bit

- Encodage par défaut
- 998 octets par ligne
- Caractères de 1 à 127
- `CR` (`x0D`) et `LF` (`x0A`) uniquement sous la forme `CRLF` (fin de ligne)

Encodages : quoted-printable

- Encodage ASCII 8 bits en caractères affichables (33-126)
- symbole ‘=‘suivi de deux chiffres en hexadécimal : `fran=E7ais`
- Retours à la ligne deviennent `CRLF`
- Lignes d’au plus 76 caractères (on utilise ‘=‘pour couper une ligne)

Encodages : base64

- 64 caractères affichables : [`A-Za-z0-9+/`]
- On traîte les octets 3 par 3 (donc 24 bits)
  - 24 bits → 4 caractères (64 possibilités = 6 bits)
  - S'il manque des octets à l'entrée on ajoute des 0
  - On ajoute des '=' à la fin pour indiquer combien de 0 ont été ajoutés

En *Python*

```
>> import base64
>> base64.b64encode('Bonjour')
'Qm9uam91cg=='
>> base64.b64decode('Qm9uam91cg==')
'Bonjour'
```

Texte brut vs. HTML

- Avantages du texte brut :
  - meilleure compatibilité
  - plus léger
  - moins de risques
- Avantages du HTML :
  - plus joli...

Formats de fichiers

Multipurpose Internet Mail Extensions

- L'ASCII 7 bits, c'est vite limité
- MIME permet d'envoyer des contenus riches :
  - Langues autres que l'anglais
  - Formats binaires (images, applications, etc.)
- La norme évolue : de nouveaux formats sont ajoutés (mais toujours version 1.0)
- Également utilisé par d'autres protocoles

Multipart

```
MIME-version: 1.0
Content-type: multipart/mixed; boundary="separation"

Ceci est un message contenant plusieurs parties au format MIME.
--separation
Content-type: text/plain

Ceci est le corps du message.
--separation
Content-type: application/octet-stream
Content-transfer-encoding: base64

U2kgdm91cyBsaXNleiBjZWNpLCBjJ2VzdCBxdWUgdm91cyBhdmV6IGJpZW4gY29t
cHJpcyBjb21tZW50IGZvbmN0aW9ubmFpdCBsJ2VuY29kYWdlIGVuIGJhc2U2NC4u
Lg==
--separation--
```

Content-type

- Également utilisé par le protocole HTTP
- Exemples : `text/plain`, `text/html`, `image/jpeg`, `image/png`, `application/octet-stream`, `video/mpeg`, etc.
- Définit le type de données, ce qui permet de les interpréter correctement

RFC et IANA

- Le problème du type de données est très répandu
- MIME Media Types `RFC 2045, 2046`
- Gérés par l'IANA
- Syntaxe : `type/sous-type`
- Types de base :
  `text`, `image`, `audio`, `video`, `application`, `multipart`, `message`

Caractères spéciaux

- En HTML, on a des entités pour écrire en US-ASCII sur 8 bits
- Problèmes évidents quand on utilise d'autres alphabets...
- Problème classique mal résolu :
  Charsets distincts

Encodage ?

- Spécifier l'encodage du texte
- Charsets IANA

```
Content-Type: text/html; charset=utf-8
Content-Type: text/html; charset=iso-8859-1
```

- Casse-têtes et transcodage à tout va

Aÿ, Faÿ-lès-Nemours, Freÿr, L'Haÿ-les-Roses et Moÿ-de-l'Aisne

iso-latin-9

- L'ISO-8859-15 c'est en gros le latin-1 + '€'

ISO-8859-15																
	x0	x1	x2	x3	x4	x5	x6	x7	x8	x9	xA	xB	xC	xD	xE	xF
0x	NUL	SOH	STX	ETX	EOT	ENQ	ACK	BEL	BS	HT	LF	VT	FF	CR	SO	SI
1x	DLE	DC1	DC2	DC3	DC4	NAK	SYN	ETB	CAN	EM	SUB	ESC	FS	GS	RS	US
2x	SP	!	"	#	$	%	&	'	(	)	*	+	,	-	.	/
3x	0	1	2	3	4	5	6	7	8	9	:	;	<	=	>	?
4x	@	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
5x	P	Q	R	S	T	U	V	W	X	Y	Z	[	\	]	^	_
6x	`	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o
7x	p	q	r	s	t	u	v	w	x	y	z	{	l	}	~	DEL
8x	PAD	HOP	BPH	NBH	IND	NEL	SSA	ESA	HTS	HTJ	VTS	PLD	PLU	RI	SS2	SS3
9x	DCS	PU1	PU2	STS	CCH	MW	SPA	EPA	SOS	SGCI	SCI	CSI	ST	OSC	PM	APC
Ax	NBSP	¡	¢	£	€	¥	Š	§	š	©	ª	«	¬	SHY	®	¯
Bx	°	±	²	³	Ž	µ	¶	·	ž	¹	º	»	Œ	œ	Ÿ	¿
Cx	À	Á	Â	Ã	Ä	Å	Æ	Ç	È	É	Ê	Ë	Ì	Í	Î	Ï
Dx	Ð	Ñ	Ò	Ó	Ô	Õ	Ö	×	Ø	Ù	Ú	Û	Ü	Ý	Þ	ß
Ex	à	á	â	ã	ä	å	æ	ç	è	é	ê	ë	ì	í	î	ï
Fx	ð	ñ	ò	ó	ô	õ	ö	÷	ø	ù	ú	û	ü	ý	þ	ÿ

http://en.wikipedia.org/wiki/ISO_8859-15

Unicode

- Table “universelle”
- 1 114 112 codes possibles (environ 250 000 assignés)
- Recouvre la plupart des encodages 8 bits
- Encodages : utf-32, utf-16, utf-8
- Nouveau standard occidental : utf-8

UTF-8

- Un à quatre octets par caractère
- Compatible avec l'`ASCII` 7 bits
- Nombre de 1 en tête du premier octet indique le nombre d'octets utilisés (les autres commencent par "`10`")
  - `0xxxxxxx` (7 bits)
    `110xxxxx 10xxxxxx` (8 à 11 bits)
    `1110xxxx 10xxxxxx 10xxxxxx` (12 à 16 bits)
    `11110xxx 10xxxxxx 10xxxxxx 10xxxxxx`

Versions

- On ajoute ou on retire des symboles à la table
- Aujourd'hui version 5.2
- Version 6.0 à venir très prochainement

Utilisation pratique

- Utiliser utf-8 comme codage
- Posséder un éditeur transcodeur
- Penser à déclarer son charset

Oui, mais...

- Comment deviner le Content-Type ?
  - par l'extension du fichier ?
  - par les méta-données du FS ?
  - en utilisant des *magic numbers* ou méta-données internes ?
- Il n'y a pas de solution miracle...

Extensions

- Heuristique couramment utilisée par les utilisateurs
- Permet de changer facilement l'application à utiliser en changeant l'extension
- Problèmes de conflits (pas de standardisation)
- Problèmes si l'on change une extension accidentellement (→ cacher les extensions → risques de sécurité !)

Méta-données

- Données stockées sur le système de fichier en dehors du fichier proprement dit
- Pas très compatible (chacun fait à sa manière)
- Difficile à transmettre (copie, courrier, etc.)

Exemple : Uniform Type Identifiers

- Méta-données de Mac OSX
- Structure hiérarchique par domaines
- Héritage de types (éventuellement multiple)
- ex : `public.png`, `com.adobe.pdf`, etc.

La magie des *magic numbers*

- Portable Network Graphics (PNG)

  `0x89 50 4E 47 0D 0A 1A 0A` → `image/png`

- JPEG

  `0xffd8` → `image/jpeg`

- Adobe PDF

  `%PDF-` → `application/pdf`

- etc...

Et le texte ?

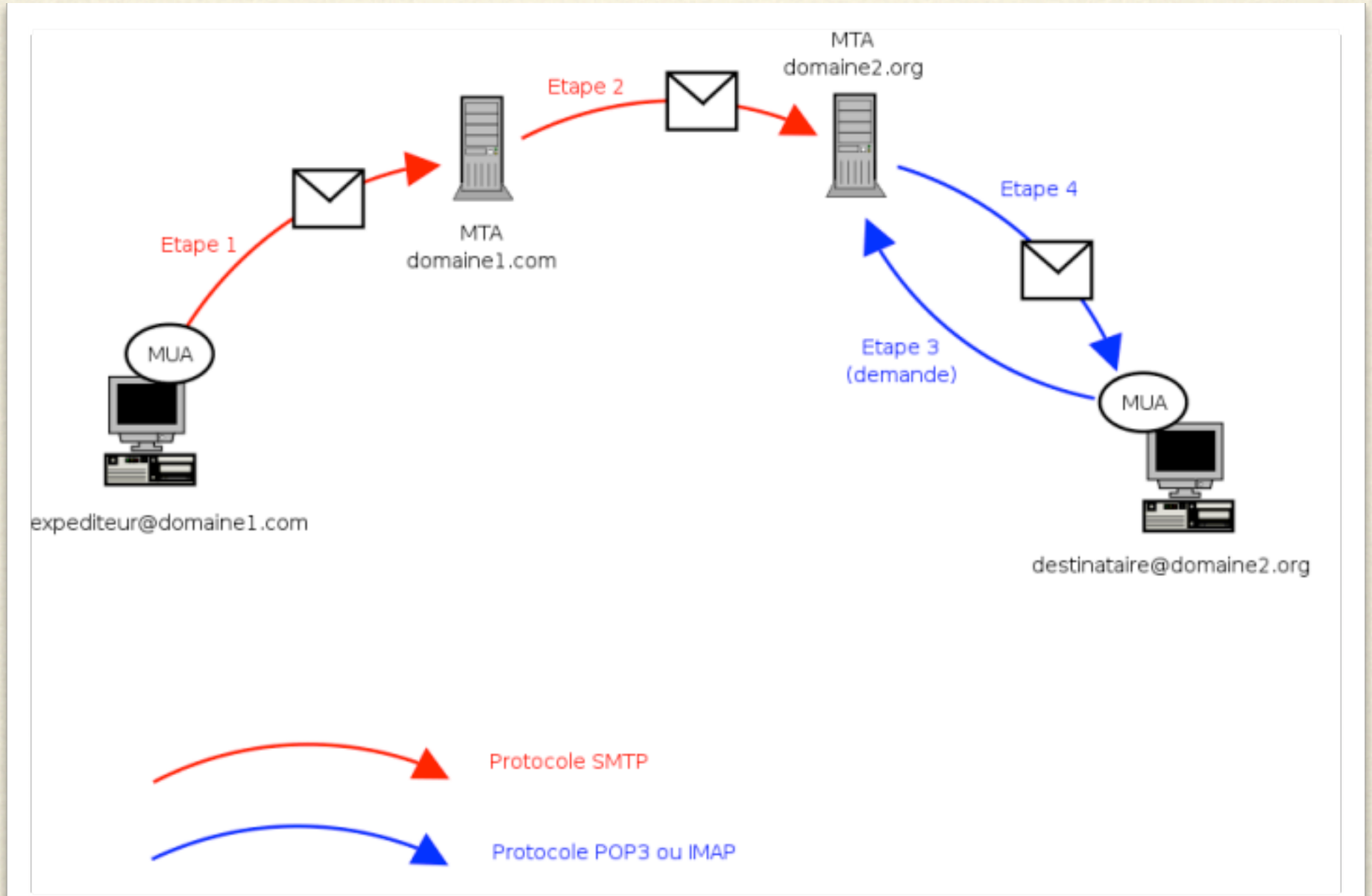
- C’est plus compliqué
- Problèmes d’encodage
- Problèmes de syntaxe libre

En pratique ?

- Déclarer le `Content-Type`
- Vérifier le `Content-Type` !
- Choisir un bon serveur, avec de bonnes heuristiques

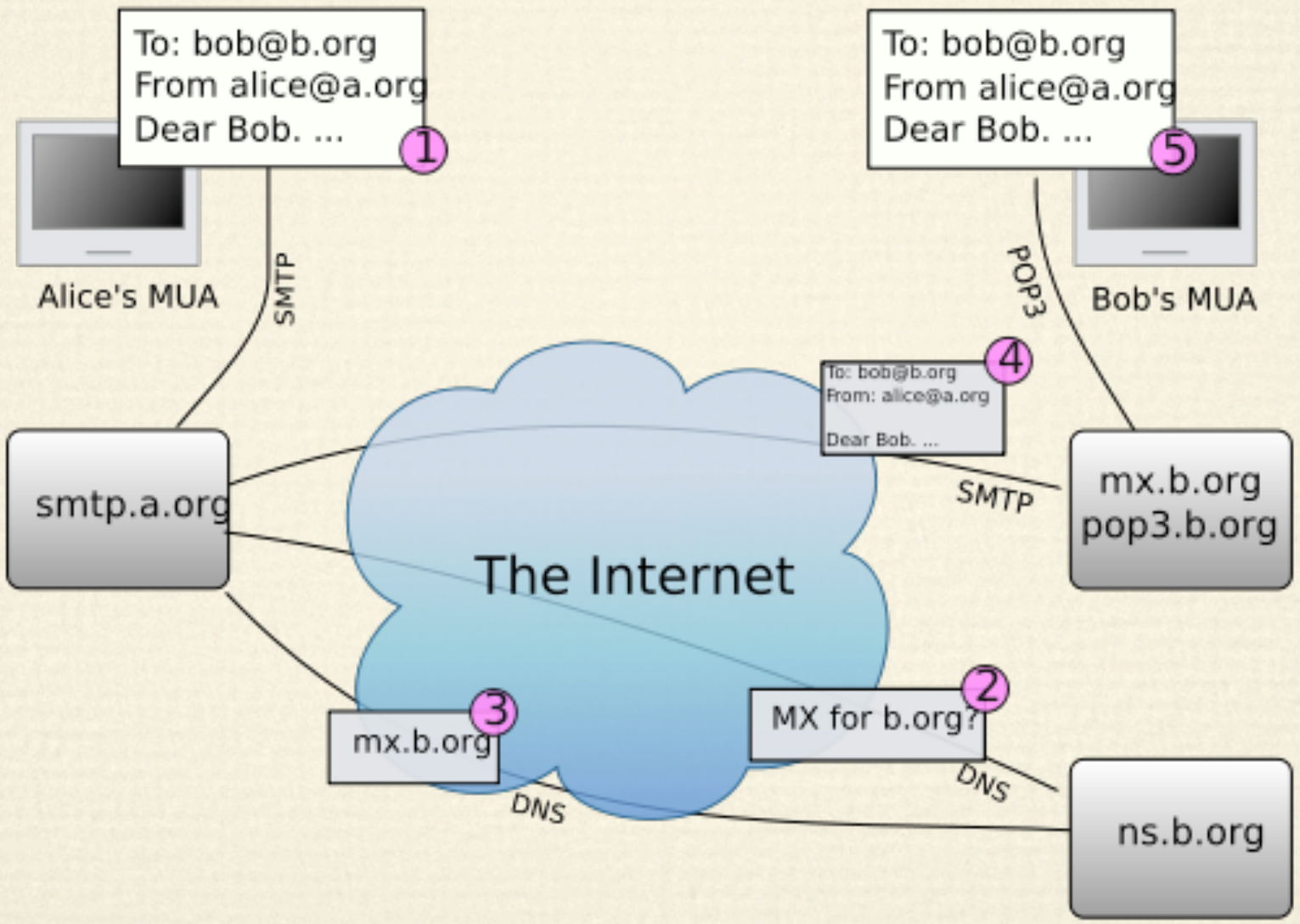
Transmission du courrier

Schéma général



source : http://fr.wikipedia.org/wiki/Serveur_de_messagerie_électronique

Schéma général (2)



source : http://en.wikipedia.org/wiki/E-mail

Client mail

- *Mail User Agent* (MUA)
- Récupère les messages sur le *Mail Delivery Agent* (MDA) et les affiche
- Encode les messages au format MIME
- Envoie les messages du client au MTA
- Se charge de l'encryption si nécessaire

Serveur mail

- Deux rôles : envoi et réception
- *Mail Transfer Agent* (MTA)
- *Mail Delivery Agent* (MDA)
- MUA → MTA → MDA → MUA
- Autres : filtres anti-spam, anti-virus, ...

Protocoles

Simple Mail Transfer Protocol

- RFC 821 (1982), 1123, 5321 (2008)
- MUA → MTA
- MTA → MTA
- MTA → MDA

Histoire

- Nombreux protocoles distincts dans les 60s
- Standards pour ARPANet (70s)
- *Mail Box Protocol* (1971), *FTP Mail* (1973), *Mail Protocol* (1973)
- SMTP se développe parallèlement à *Usenet* (alternative à UUCP)
- *Sendmail* (début 80s) : premier serveur SMTP

SMTP : Caractéristiques

- Echange de messages textuels
- Adresses des destinataires + message [+ objets encodés dans le message]
- Chaque MUA connaît un MTA (configuration)
- Connexion par TCP sur le port 25 (par défaut)
- Protocole *push-only*

Open Mail Relay

- Un serveur mail qui transmet tous les messages
- Initialement le cas de tous, aujourd'hui peu utilisé (risques de spam, donc liste noire)
- Uniquement messages d'adresses IP autorisées (FAI) ou d'utilisateurs authentifiés (SMTP-AUTH)

Store and Forward

- Les messages passent par des intermédiaires
- Chaque intermédiaire essaie de transmettre le message à un noeud plus proche de la destination
- Fortement asynchrone
- Forte tolérance aux pannes

```
S: 220 smtp.example.com ESMTP Postfix
C: HELO relay.example.org
S: 250 Hello relay.example.org, I am glad to meet you
C: MAIL FROM:<bob@example.org>
S: 250 Ok
C: RCPT TO:<alice@example.com>
S: 250 Ok
C: RCPT TO:<theboss@example.com>
S: 250 Ok
C: DATA
S: 354 End data with <CR><LF>.<CR><LF>
C: From: "Bob Example" <bob@example.org>
C: To: Alice Example <alice@example.com>
C: Cc: theboss@example.com
C: Date: Tue, 15 Jan 2008 16:02:43 -0500
C: Subject: Test message
C:
C: Hello Alice.
C: This is a test message with 5 headers and 4 lines in the body.
C: Your friend,
C: Bob
C: .
S: 250 Ok: queued as 12345
C: QUIT
S: 221 Bye
```

source : http://en.wikipedia.org/wiki/SMTP

Extended SMTP (ESMTP)

- IETF RFC 1869 (en 1995)
- `HELO` → `EHLO`
- Étend le protocole SMTP et permet d'ajouter des extensions
- Authentification (SMTP-AUTH)

Mail eXchanger (MX)

- Enregistrement DNS
  - Nom de domaine → liste de serveurs SMTP
  - Un MTA contacte le DNS du domaine cible et envoie le message au MTA correspondant

Mail eXchanger : priorités

- Les listes de MTA sont ordonnées par priorité
- Le MTA essaie de contacter les serveurs selon ces priorités jusqu'à obtenir une réponse
- S'il n'y a pas d'entrée MX, utiliser l'entrée A (IP correspondant au nom de domaine)

Utilité des priorités

- SMTP : *Store and Forward*
- En cas d'échec, délai avant prochain essai
  - Machines secondaires "savent" quand le serveur est de retour (mieux que de ne pas transférer le message)
  - Mais il est plus coûteux d'utiliser les machines secondaires que la machine primaire

MX et le spam

- Certains MTA de spam ne sont pas conformes à la norme
- Ils utilisent uniquement le premier MTA de la liste MX ou uniquement le dernier
- On peut esquiver ça en plaçant des leurres en première et dernière position
- Ne fonctionne que contre les serveurs de spam très simples...

Post Office Protocol v.3 (POP3)

- Client interroge le serveur pour obtenir les messages
- Adapté aux connexions courtes (téléchargement des messages, suppression sur le serveur, consultation hors-ligne)
- TCP/IP port 110
- Plusieurs utilisateurs peuvent accéder à la même boîte mail

Sécurité

- Initialement aucune encryption*
- APOP : envoi du mot de passe sous forme de *hash*
- On peut également utiliser une connexion TLS/SSL

(*) ce mot n'existe pas en français mais il est bien pratique...

Identification des messages

- Les messages sont numérotés sur le serveur
- Les numéros changent quand on supprime des messages
- On associe un identifiant unique (UID) à chaque message
- Table de correspondance UID ↔ numéro
- Peut devenir lourd sur de très grande boîtes de réception

Quelques commandes

- `USER`, `PASS`, `APOP` : identification
- `LIST` : liste des messages (avec la taille)
- `STAT` : nombre de messages, taille totale
- `RETR`, `TOP` : obtenir un message ou une portion
- `DELE` : supprimer un message

D’autres commandes

- `NOOP` : ne rien faire
- `QUIT` : quitter la session
- `RSET` : réinitialiser la session
- `UIDL` : obtenir les identifiants uniques d’un ou plusieurs messages

```
S:    +OK POP3 server ready <1896.697170952@dbc.mtview.ca.us>
C:    APOP mrose c4c9334bac560ecc979e58001b3e22fb
S:    +OK mrose's maildrop has 2 messages (320 octets)
C:    STAT
S:    +OK 2 320
C:    LIST
S:    +OK 2 messages (320 octets)
S:    1 120
S:    2 200
S:    .
C:    RETR 1
S:    +OK 120 octets
S:    <the POP3 server sends message 1>
S:    .
C:    DELE 1
S:    +OK message 1 deleted
C:    RETR 2
S:    +OK 200 octets
S:    <the POP3 server sends message 2>
S:    .
C:    DELE 2
S:    +OK message 2 deleted
C:    QUIT
S:    +OK dewey POP3 server signing off (maildrop empty)
```

source : http://en.wikipedia.org/wiki/Pop3

POP4 ?

- Travail en cours (mais pas bcp de progrès récemment)
- Gestion de répertoires
- Messages en plusieurs parties

Internet Message Access Protocol

- Alternative à POP3
- Aujourd'hui IMAP v.4 rev.1(RFC 3501)
- TCP/IP port 143

Caractéristiques

- Deux modes : en ligne et hors ligne
- Messages sont conservés sur le serveur (sauf suppression explicite)
- La plupart des clients mail supportent IMAP, mais pas tous les FAI
- Plusieurs utilisateurs peuvent accéder à la même boîte mail (mais pas en même temps)

Histoire

- Créé en 1986 par Mark Crispin
- But : gestion distante de boîte mail (par opposition au fonctionnement de POP)
- Version actuelle : 1996

Sécurité

- Cryptage natif (plusieurs méthodes d'authentification sécurisée, le client et le serveur se mettent d'accord)
- Possibilité d'utiliser TLS/SSL (IMAPS port 993 ou utilisation de la commande `STARTTLS` en cours de session)

```
* OK [CAPABILITY IMAP4REV1...
001 login testimap2 testimap2
001 OK [CAPABILITY IMAP4REV1 ...
002 select INBOX
* 1 EXISTS
* NO Trying to get mailbox lock from process 28032
* 0 RECENT
* OK [UIDVALIDITY 1068367935] UID validity status
* OK [UIDNEXT 2] Predicted next UID
* FLAGS (\Answered \Flagged \Deleted \Draft \Seen)
* OK [PERMANENTFLAGS (\* \Answered \Flagged \Deleted \Draft \Seen)]
Permanent flags
002 OK [READ-WRITE] SELECT completed

003 fetch 1 BODY[HEADER]
* 1 FETCH (BODY[HEADER] {471}
  <l'en-tête du message>
)
003 OK FETCH completed

004 fetch 1 BODY[TEXT]
* 1 FETCH (BODY[TEXT] {8}
  <le corps du message>
)
004 OK FETCH completed
005 logout
* BYE gw2.maison.mrs IMAP4rev1 server terminating connection
005 OK LOGOUT completed
```

source : http://stielec.ac-aix-marseille.fr/cours/caleca/imap/le_protocole.htm

Avantages d'IMAP sur POP3

- Mode *en-ligne* permet d'exploiter une connexion persistante
- Plusieurs connexions simultanées à une boîte mail
- Possibilité de traîter séparément les parties d'un message (format MIME)

Avantages (2)

- Utilisation de marqueurs (*flags*) pour les propriétés d'un message (lu, répondu, supprimé, etc.) sur le serveur
- Gestion de boîtes multiples (structure de répertoires)
- Recherche côté serveur (système de requêtes)
- Possibilité d'élargir le protocole à l'aide d'extensions (sous un format pré-défini)

Inconvénients

- Plus grande charge sur le serveur (recherche, gestion des accès concurrents, ...)
- Nécessité de faire plus de requêtes (problème sur connexions très limitées)
- Un message envoyé en SMTP doit être transmis une seconde fois pour être stocké dans le répertoire *éléments envoyés* (fonctionnalité inexistante en POP3...)

Webmail

- Une application web
- Accès à une boîte mail par un navigateur
- Exemples : *Gmail*, *Yahoo! mail*, *Hotmail*, etc.
- Permet l'accès de toute machine connectée sans installation
- Mais pas de mode hors-ligne…