Faculté des Sciences & Techniques — Université de Limoges

Master 1ère année

Protocoles & Prog. Réseau

TP n°3

Envoi de courrier avec pièce attachée

Le protocole SMTP, *Simple Mail Transfer Protocol*

Pour envoyer un courrier, le client se connecte en TCP au serveur SMTPet dialogue avec lui suivant le protocole SMTP.
Ce protocole est composé de l'envoi d'une commande ("`HELO`", "`MAIL FROM:`", "`RCPT TO:`", "`DATA`", ...) suivie de paramètres, chaque commande étant terminée par un « `\n` » ou « `\r\n` ».

Exemple de transaction entre le client et le serveur suivant le protocole SMTP :

Description	ligne envoyée par le Client	ligne reçue depuis le Serveur
Connexion du client au serveur		
Invite du serveur		220 smtp.unilim.fr
Présentation du client	HELO client.unilim.fr	
Salutation du serveur		250 Nice to meet you
Indication de l'expéditeur	MAIL FROM: <alice@unilim.fr>	
Acceptation de l'expéditeur		250 Sender ok
Indication du destinataire	RCPT TO: <bob@unilim.fr>	
Acceptation du destinataire		250 Recipient ok
Début du courrier	DATA	
Acceptation		354 Enter mail end with "." on a line by itself
Contenu du courrier	bla bla bla	
Fin du contenu du courrier	.	
Validation du courrier		250 Message accepted for delivery
Fin de la connexion du client	QUIT	
Fin de la connexion du serveur		

Commentaires :

- commande "`HELO`" : le client donne en paramètre l'adresse symbolique de la machine depuis laquelle il se connecte.
  Ces informations sont enregistrées dans les fichiers de rapports d'activité du serveur (fichier « log »).
- commandes "`MAIL FROM:`" et "`RCPT TO:`" : le client donne en paramètre l'adresse électronique de l'expéditeur et du destinataire ;
  Ces adresses sont éventuellement vérifiées par le serveur SMTP afin d'en vérifier la validité (éviter les courriers anonymes, les courriers "Spam"…)
- le serveur confirme au client qu'il accepte le traitement du courrier ;
- commandes "`DATA`" et "`.`" : le client envoie le contenu du courrier et finit par une ligne contenant seulement un « . » au début.

Question

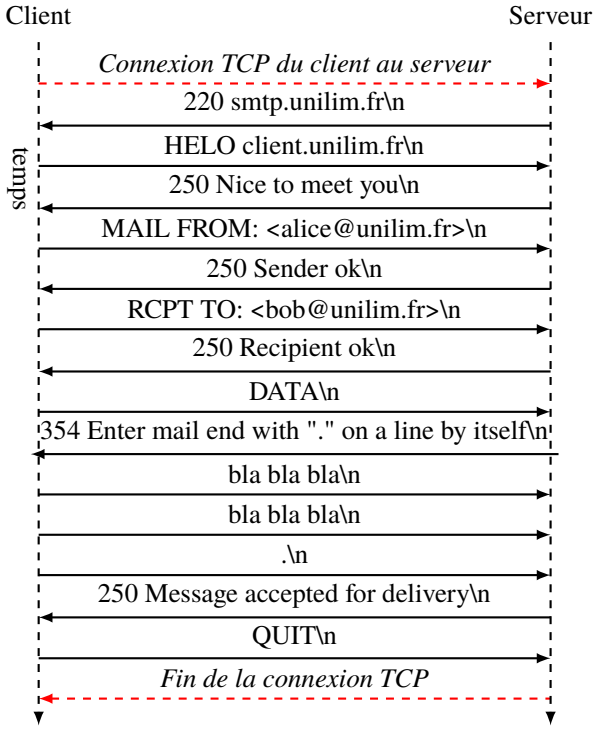
1 – En s'inspirant de l'exemple précédent, écrire un programme Python qui :

a. demande à l'utilisateur les adresses du destinataire et de l'expéditeur (des valeurs par défaut peuvent être proposées) ;

b. permet de saisir le contenu du courrier ;

c. envoie le courrier à l'aide du protocole SMTP en se connectant en TCP à un serveur SMTP. *Pour cette connexion, il faut utiliser le port associé au service SMTP, c-à-d le port 25, et pour l'adresse du serveur, utiliser par exemple la machine* `smtp.unilim.fr`.

MSC, « *Message Sequence Chart* » du protocole SMTP



Gestion des codes d'erreur/information du protocole SMTP

Le premier chiffre indique le succès ou non de l'opération :

- ▷ « 2 » signifie *Success* (succès) : l'opération s'est déroulée sans problème ;
- ▷ « 3 » signifie *Information* : ce sont des lignes qui renseignent sur l'état du serveur ou sur une étape, et qui peuvent être ignorées ;
- ▷ « 4 » indique une *Persistent Transient Failure* (erreur temporaire) : vous devriez réussir à envoyer le même message un peu plus tard sans rien modifier ;
- ▷ « 5 » signale une *Permanent Failure* (erreur permanente) : il faut modifier le message ou ses entêtes pour réussir à l'envoyer.

Question

2 – Améliorer le programme précédent pour prendre en compte les lignes d'erreur.

Amélioration du contenu du courrier

Le format MIME, "*Multi-purpose Internet Mail Extension*", a été défini pour améliorer le courrier en permettant d'y joindre des fichiers binaires (programme, photos…).
Ces fichiers binaires sont appelés « pièces jointes » ou « pièces attachées ».

Ce format est une **façon de formater** le contenu du courrier suivant une **structure rigide** qui permet d'associer aux données transmises :

- ⋆ un **type** ;
- ⋆ un **encodage** (pour dépasser le format limité à 7 bits du protocole SMTP, RFC 821) ;
- ⋆ …

Dans cette **structure rigide**, il est possible de définir au moins deux parties :

i. l'une correspondant au **contenu du courrier** au format texte ;
ii. l'autre correspondant au **contenu du fichier binaire** encodé en texte (en base64 par exemple).

Pour cela on donne au contenu du courrier un type « fourre-tout », le type "*multipart/mixed*", qui permet d'inclure différentes sous-parties (ou zones) :

- ▷ chacune des sous-parties est séparée par une frontière (une chaîne de texte qui **ne peut être rencontrée** dans le reste du courrier) ;
- ▷ pour chacune de ces sous-parties, on inclut **une entête MIME** pour décrire le contenu de cette sous partie.

Nouveau contenu de courrier avec le format MIME :

Entête	`MIME-Version: 1.0\n`
	`Content-type: multipart/mixed; boundary="AZERTYUIOP";\n`
	`Subject:` *titre du courrier*`\n`
1ère frontière	`--AZERTYUIOP\n`
entête de zone	`Content-type: text/plain\n`
ligne vide	`\n`
contenu du courrier	*bla bla bla*`\n`
2nde frontière	`--AZERTYUIOP\n`
entête de zone	`Content-type: octet/stream; name='monfichier';\n`
	`Content-transfer-encoding: base64\n`
ligne vide	`\n`
pièce jointe	*le code base64*`\n`
dernière frontière	`--AZERTYUIOP--\n`

Remarques :

▷ Les retours à la ligne sont ici des « \n », mais peuvent être des « \r\n ».

▷ Le contenu MIME s'insère dans le protocole SMTP vu précédemment, c-à-d entre les commandes `DATA` et la ligne ne contenant qu'un « `.` ».

Explications du format MIME

Il existe en fait une seule zone principale, dont la nature est de contenir plusieurs zones (celle dont le type est "multipart/mixed") :

⋆ chaque zone est repérée par une en-tête indiquant :
  ◇ la nature de la zone (image, son, texte…) : `text/plain`, `text/html`, `image/jpeg`…
  ◇ la représentation choisie (purement textuelle, ou encodage de données binaires…).

⋆ chacune de ces zones est séparée par une frontière définie au début "boundary" ;

⋆ chaque frontière est précédée de « −− » ;

⋆ la dernière frontière est également suivie de « −− ».

Attention

Si le format MIME n'est pas *scrupuleusement* suivi dans le courrier :

⋆ son contenu n'est **pas reconnu** comme du MIME, c-à-d il n'existe plus de notion de pièces jointes ;

⋆ l'utilisateur voit l'intégralité du contenu du courrier, y compris les caractères de l'encodage base64 (ou bien ne voit rien du tout !).

Question

3 – Reprendre le programme précédent afin de pouvoir envoyer une pièce attachée encodée au format MIME :

a. demander à l'utilisateur le nom du fichier binaire à transmettre ;

b. transmettre la structure MIME :
  ◇ la partie contenant le courier;
  ◇ la partie contenant l'encodage base64 du fichier binaire (utilisez le module base64 de Python).

Utilisation du module `base64`

```
import base64
>>> a=base64.encodebytes(b'toto')
>>> a
b'dG90bw==\n'
>>> base64.decodebytes(a)
b'toto'
```

Remarques :

◦ Pour vérifier que le courrier envoyé est bien structuré, utilisez l'outil de courrier « Thunderbird » : si votre courrier est correct, il le traitera *correctement* et vous affichera les pièces jointes !

◦ Pour faciliter le traitement de la pièce jointe par l'outil de courrier, il vaut mieux choisir une image (type MIME « `image/jpeg` » par exemple), qui sera affichée directement dans le contenu du courrier.