Chapitre IV Partie 5 Protocoles d'echange Mail (SMTP, POP et IMAP)- 2

Ilyas Bambrik

Table des matières

Post Office Protocol (POP)

I

- Après la livraison du courrier au serveur mail approprié, l'utilisateur concerné par le courrier pourra le voir lors de son prochain accès à ça boite mail. L'accès au mail se fait par un protocole indépendant à SMTP. Les deux protocoles d'accès aux mails les plus utilisés sont *POP (Post Office Protocol)* et IMAP (Internet Message Access Protocol).
- La première version de POP (*POP3 est la version courante de ce protocole*) permet simplement à un utilisateur de :

    1. Transmettre son nom et mot de passe pour s'authentifier ;
    2. Ensuite, l'utilisateur peut accéder à ça boite mail et télécharge ses mails (le contenu de la boite complète) ;

- Similairement au serveur SMTP, un serveur POP fonctionne 24/7 pour recevoir les demandes de connexion utilisateurs et les servir. Ce type de serveur écoute *généralement* le port *TCP 110* ;
- La dépendance entre POP et SMTP est claire. POP permettra d'accéder aux mails reçu par le serveur SMTP. Ainsi, le serveur SMTP et le serveur POP peuvent être hébergés dans la même machine. Ou bien, le serveur POP possède un accès par réseaux au dépôt des mails SMTP ;

Commandes et réponses POP

II

- Les commandes POP sont de 3 à 4 lettres ASCII comme FTP et SMTP. Par contre les codes réponses sont de deux types seulement : *+OK* (*commande exécutée avec succès*) et *-ERR* (*échec d'exécution de la commande*).
- Chaque commande / réponse échangée entre le serveur et client est terminée par *CRLF* ;
- La session POP commence par une connexion TCP entre le client et le serveur sur le port dédié *110 TCP*.

1. Authetification

1. Immédiatement après la connexion, le serveur transmet un message du genre : "*+OK Howdy !*" pour indiquer au client qu'il peut commencer l'échange avec le serveur ;
2. Le client s'authentifie ensuite d'une manière identique à FTP (envoie de la commande *USER* avec le nom de l'utilisateur et ensuite *PASS* avec le mot de passe) ;
3. Si l'utilisateur s'authentifie correctement, le serveur répondra avec un code +OK et un message ;

Syntaxe : Exemple d'authentification IMAP

```
+OK Howdy! we are at your service!
USER Ilyas@boitemail.com
+OK Waiting for password
PASS CodingMonk?L7*5
+OK Ilyas@boitemail.com has 9999 messages
```

2. Commande d'échange avec le serveur

Les commandes les plus importantes pour gérer et accéder à la boite mail sont les suivantes :

1. *STAT* : renvoie le nombre de mails dans la boite et la taille totale occupé par celle-ci ;
2. *LIST* : renvoie une liste contenant le numéro de chaque mail accompagné avec ça taille ;
3. *RETR* : récupère le mail désigné par le numéro en paramètre sous format RFC 822 (*voir le cours SMTP*). Par exemple: après l'envoie "*RETR 1090CRLF*" le serveur répond avec le contenu du mail 1090 si celui-ci existe ;
4. *DELE* : marque le mail désigné *pour être supprimé du serveur* à la fin de la session (le message désigné par cette commande ne sera pas immédiatement supprimé, voir la dernière commande *QUIT*);
5. *NOOP* : teste si la connexion avec le serveur est toujours vivante ;
6. *RSET :* réinitialisation de la session (les suppressions des messages par DELE seront annulées) ;

7. *TOP :* renvoie les entêtes du message désigné et les N premières lignes du message ; "TOP 1 10CRLF" renvoie l'entête du message en plus des 10 premiers lignes du contenu du message (pas touts les serveurs mails supportent cette commande);
8. *UIDL* : retourne la liste des identificateurs uniques des messages dans la boite ;

La dernière commande à transmettre au serveur est la commande *QUIT*. Celle-ci indique la fin de la session *ainsi que confirme la suppression des mails marqués par le commande DELE*. Avant l'envoie de *QUIT*, l'utilisateur peut annuler les suppressions par *RSET*;

Si une déconnexion entre le client et le serveur se produit sans que le client envoie "*QUIT*", les messages marqués par *DELE* ne seront supprimés ;

Le modèle d'opération de POP est *offiline* car l'utilisateur récupère ses messages sur ça machine locale et *ensuite typiquement supprime le message déjà lu du serveur*.

Cependant, POP ne permet pas de marquer les messages lus ou non lus par exemple, non plus de regrouper la boite mail sur plusieurs catégories/ répertoires (spams, news, social medias, work etc). Ce genre de fonctionnalités est assuré par un protocole d'accès aux mails plus sophistiqué appelée IMAP (*Internet Message Access Protocol*);

Remarque

Comme dans le format MIME, le serveur signale la fin d'une réponse par un *"."* au début de la ligne ;

```
STAT
+OK 2 574
LIST
+OK
1 414
2 160
.
RETR 1
+OK
(Message 1 is sent)
.
DELE 1
+OK message 1 deleted
RETR 2
+OK
(Message 2 is sent)
.
DELE 2
+OK message 2 deleted
QUIT
+OK Bye
```

Internet Message Access Protocol (IMAP)

III

- Similairement à POP, un serveur IMAP permet au client d'accéder à ça boite de mail. Afin d'accomplir ceci, le serveur IMAP doit accéder aux mails livrés par SMTP (*comme expliqué dans la partie POP3*) ;
- Un serveur IMAP écoute le port TCP/143 ;

1. Commandes et réponses IMAP

- Après la connexion du client au port *TCP/143* , le serveur *IMAP* généralement envoie un message avec le code **OK* (comme le +OK de POP) ;
- Dans certaines situations, le serveur peut reconnaître l'identité du client dès le début de la session. Dans une telle situation le serveur répond avec le code **PREAUTH* avec le nom de l'utilisateur pré-authentifié. Cette fonctionnalité est appelée Pré-authentification et dans un tel cas, l'utilisateur n'a pas besoins de s'authentifier par un nom utilisateur et mot de passe ;
- En cas où le serveur n'accepte pas une connexion ou clôture la session, le réponse **BYE* est transmise vers le client ;
- Similairement à POP, en cas d'échec d'une commande le serveur répond avec **NO*. Par contre, les réponses indiquant une erreur sont distinguées par le code **BAD* ;
- Comme POP, les commandes IMAP sont délimitées par CRLF. Par contre, chaque commande transmise par le client doit avoir un code unique (généralement il s'agit d'un code ascendant). Ainsi, la réponse du serveur comprendra l'identifiant de la commande correspondante. De plus, ceci permet l'envoie de plusieurs commandes sans attendre les réponses de commandes précédentes (*par contre, ce-ci ne s'applique pas à toutes les commandes)* ;

Les commandes IMAP

IV

Remarque

- Contrairement à POP, IMAP est conçu pour permettre de regrouper / gérer l'espace de l'utilisateur sur plusieurs boites mails (*répertoire*). Ainsi, après l'authentification, au lieu que l'utilisateur n'accède directement à ses mails (comme dans POP) , celui-ci sélectionne la boite mail à la quelle il souhaite accéder ;
- La spécification détaillée de IMAP dit que l'utilisateur a accès à plusieurs boites mails, mais en réalité, l'utilisateur possède une seule boite organisée sur plusieurs répertoires (*folders*).

Exemple :

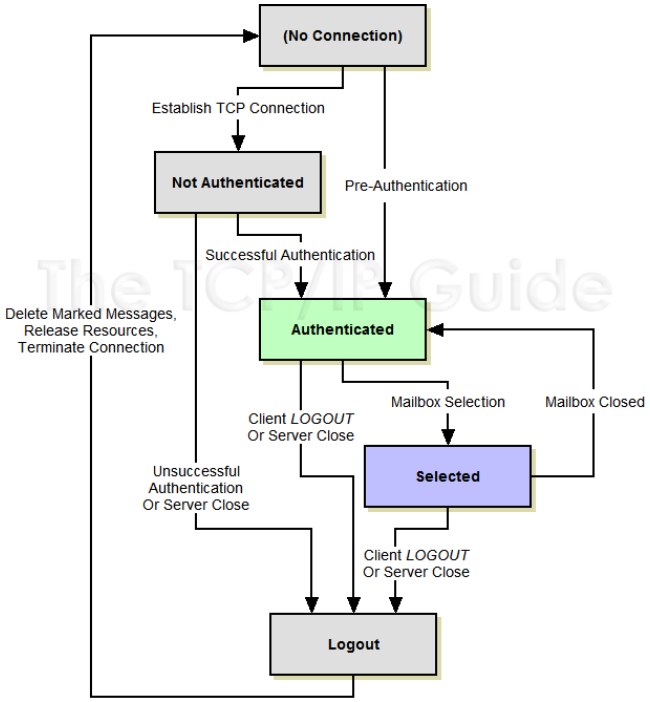
L'utilisateur propriétaire de *utilisateur@boitemail.com* accède à ça boite après authentification. Et ensuite, il a le choix entre plusieurs boites : "SPAM", "INBOX", "URGENT", etc. Dans l'étape de la sélection, l'utilisateur fait le choix entre les différents répertoires (et non pas une autre boite du genre *utilisateur.MABOITE2@boitemail.com*) ;

Les commandes IMAP sont réparties sur trois groupes :

- *Commandes "Any state" :* ce type de commande peut être exécuté à n'importe quel moment.
    - *CAPABILITY :* pour avoir les fonctionnalités supportées par le serveur ; *NOOP*: pour vérifier si la connexion avec le serveur est toujours valable ; *LOGOUT* : pour quitter la session;
- *Commandes d'authentification :* permettant de s'authentifier ;
- *Commandes après authentification :* généralement, ce genre de commande permet de sélectionner une boite mail ;
- *Commandes après sélection :* permettent d'accéder / gérer les mails situés dans une boite ;

par ces commandes, l'utilisateur transite entre les états suivants :

1. *Non-Authentifié* : l'utilisateur doit s'authentifier pour passer à l'état "(2) *Authentifié*" et accéder à son espace;
2. *Authentifié* : Dans cet état, l'utilisateur doit sélectionner la boite /répertoire au quel il souhaite accéder ;
3. *Boite/répertoire sélectionnée* : Une fois que la boite est sélectionnée, l'utilisateur peut accéder et gérer les mails contenus dans la boite selectionée ;
4. *Déconnecté* : Une fois que l'utilisateur transmet la commande LOGOUT, celui-ci se déconnecte de la session ;



1. Commandes d'authentification

- Traditionnellement, l'utilisateur commence par s'authentifier par la commende *LOGIN*. Contrairement à l démarche d'authentification POP, le mot de passe et le nom d'utilisateur sont transmis dans une seule ligne en argument de la commande *LOGIN* ;
- En cas de réussite d'authentification, le serveur répond par **OK* et un message textuel. Sinon, le serveur renvoie le code **NO* pour dire que le processus a échoué ;

Syntaxe

```
*OK Server ready to receive commands
1 LOGIN nomutilisateur@boitemail.com motdepasseutilisateur
1 OK nomutilisateur@boitemail.com logged in successfully
2 LOGOUT
* BYE tethys.ringofsaturn.com IMAP4rev1 server terminating connection
2 OK LOGOUT completed
```

- Dans l'exemple précédant, l'utilisateur fournie le nom d'utilisateur et mot de passe sur la même ligne avec la commande LOGIN (*nomutilisateur@boitemail.com motdepasseutilisateur*) ;
- La première commende *"LOGIN nomutilisateur motdepasseutilisateur"* possède le tag *"1"* (préfixe). La réponse correspondante commence par le même tag (*1*). La même chose pour la commande *"2 LOGOUT"* (*tag =2*) ;

2. Commande après authentification :

Après l'authentification, l'utilisateur peut gérer ces boites mail (mais ne peut pas accéder directement aux mails avant de spécifier à quelle boite mail il souhaite accéder). Les commandes les plus importantes de cette étape sont :

- *SELECT* : Permet de sélectionner une boite mail. Cette commande a pour paramètre le nom de la boite que l'utilisateur souhaite accéder.
- *EXAMINE* : Fonctionne exactement comme *SELECT* sauf que l'accès à la boite est en mode de lecture seul (aucun changement à la boite ne peut être fait) ;
- *CREATE* : permet de créer une nouvelle boite mail ;
- *DELETE* : Supprime la boite mail avec le nom fournie en paramètre ;
- *RENAME* : Permet de renommer une boite mail ("*RENAME NOMBOITE NOUVELNOMBOITE*" renomme la boite *NOMBOITE* à *NOUVELNOMBOITE*) ;
- *LIST* : Revoie une liste des noms des boites mails qui corresponds partiellement ou complètement à la chaîne de caractères en paramètre ;
- *STATUS* : Renvoie des informations (comme le nombre de mails non lus dans la boite) sur la boite mail avec le nom passé en paramètre ;
- *APPEND* : Ajoute un message à une boite mail ;

Si la commande *SELECT* ou *EXAMINE* s'exécute correctement (la boite sélectionnée par le client existe), le serveur répondra avec les informations suivantes concernant la boite sélectionné :

1. *EXISTS*: Indique le nombre total des mails dans la boite ;
2. *RECENT* : Le nombre de mails récemment reçu ;
3. *FLAGS* : Flags des mails supportés par le serveur (généralement : *"\Seen", "\Answered", "\Flagged", "\Deleted", "\Draft" et "\Recent"*) ;
4. *UNSEEN (information optionnelle)*: le numéro du premier (le plus ancien) mail qui n'a pas été lu ;
5. *PERMANENTFLAGS (information optionnelle)*: Liste des flags que l'utilisateur peut changer ;
6. *UIDNEXT* : Identifiant unique anticipé du prochain message reçu dans cette boite ;
7. *UIDVALIDITY* : Identifiant unique du répertoire dans cette session courante ;

Complément

Les flags qui marque l'état d'un mail ne permette pas seulement à un client de savoir quel message a été lu (*"\Seen"*) et quel message vient d'être reçu (*"\Recent"*). Ce système permet aussi l'accès aux mails par différents utilisateurs sans que l'un interfère avec les autres.

Exemple : les utilisateurs peuvent accéder au mails / répertoire de la même boite

- *UIDNEXT* permet a un client de vérifier si un nouveau message a été reçu depuis ça dernière connexion .
  Exemple : Si la valeur de *UIDNEXT* augmente par rapport à l'accès précédant, ce-ci signifie qu'un nouvel mail a été reçu ;

- *UIDVALIDITY* permet au programme client de vérifier si une boite / répertoire a été modifier par un autre utilisateur (dans ce cas) ;
  Exemple : Si un utilisateur supprime la boite avec le nom '*INBOX*' et crée une nouvel boite avec le même nom ('*INBOX*'), les autres utilisateurs qui accéderont par la suite à '*INBOX*' remarqueront le changement du *UIDVALIDITY* depuis le dernier accès.

3. Commandes après sélection de la boite

Une fois que la boite est sélectionnée, l'utilisateur peut effectuer des opérations sur les messages de la boite :

- *SEARCH* : permet de chercher dans la boite sélectionnée selon le critère en paramètre ;
- *STORE* : Change les flags du message en paramètres ;
- *FETCH* : permet de récupérer un ou plusieurs messages de la boite mail. Cette commande peut être utilisée pour récupérer une partie du message (Date, Enveloppe, Texte, etc) ;
- *COPY* : copie une ou plusieurs messages à la boite mail spécifiée en paramètre;
- *EXPUNGE* : supprime touts les message qui ont été marqué avec le flag *"\Deleted"* ;
- *CLOSE* : Ferme la boite courante pour que l'utilisateur peut choisir une autre. Comme dans POP, CLOSE opère de la même manière que *QUIT* : quand l'utilisateur envoie la commande *CLOSE* touts les messages marqués par un flag *\Delete* seront supprimés;

Syntaxe : FETCH

Les exemples suivant montrent la syntaxe de la commande FETCH :

```
1 TAG001 FETCH 1 full # le serveur transmettra le contenu complet du message 1
2 TAG002 FETCH 4 BODY[] # le serveur transmettra le contenu complet du message 4
  (fonctionne comme le premier exemple)
3 TAG003 FETCH 5 FLAGS # le serveur transmettra les flags du message 5
4 TAG004 Fetch 10 (BODY.PEEK[HEADER]) # Renvoie de l'entête du message seulement
5 TAG005 FETCH 1 BODY.PEEK[TEXT] # Renvoie du contenu du message seulement
6 TAG006 FETCH 1 RFC822.SIZE # Renvoie de la taille du message seulement
```

Syntaxe : STORE

TAG001 STORE 20 +FLAGS (\Deleted \Seen) # ajoute le flag \Deleted et \Seen au message 20

TAG002 STORE 1 -FLAGS (\Deleted \Seen) # supprime le flag \Deleted et \Seen de la liste des flags du message 1

TAG003 STORE 30 FLAGS \Answered # toute la liste des flags précédants du message 30 seront remplacés par le flag \Answered

Syntaxe : Commande SEARCH

```
4 SEARCH FROM notifications@github.com

* SEARCH 1789 1791

4 OK SEARCH Completed

5 SEARCH TO support@cybrary.it

* SEARCH 1900

5 OK SEARCH Completed
```

- La séquence de commandes prétendante *avec tag ==4* permet de rechercher les messages reçus de la part *notifications@github.com* (en résultat on a obtenu le message numéro 1789 et 1791). La séquence avec tag==5, permet de rechercher les mails adressés à *support@cybrary.it*.
- Pour rechercher les mails correspondant à un Flag *\SEEN* : SEARCH SEEN
- La recherche peut se faire par contraintes de date :

```
TAG001 SEARCH SENTBEFORE 12-Mar-2016 # reçu avant 12-Mar-2016
TAG002 SEARCH SENTON 12-Mar-2016 # reçu le jour 12-Mar-2016
TAG003 SEARCH SENTSINCE 12-Mar-2016 # reçu depuis 12-Mar-2016
TAG006 SEARCH BEFORE 12-Mar-2016 # reçu avant 12-Mar-2016
TAG007 SEARCH YOUNGER 3600 # reçu il y a moins d'une heure (3600sec)
TAG008 SEARCH OLDER 3600 # reçu il y a plus d'une heure (3600sec)
```

La syntaxe des conditions complexes est une syntaxe *postfixée*

```
SEARCH OR (FROM notification@github.com) (FROM issues-reply@bitbucket.org) #
message de la part de notification@github.com ou issues-reply@bitbucket.org
----------------------------------------------------------------------
SEARCH NOT (OR (FROM notification@github.com) (FROM issues-reply@bitbucket.org))
# message ne provenant pas ni de notification@github.com ni de issues-reply@bitbucket.
org
----------------------------------------------------------------------
```

Complément : Fetch

Le serveur marquera automatiquement un message avec le flag *\Seen* si son contenu a été récupéré par l'utilisateur (avec *FETCH*) *et que celui-ci a accédé à la boite mail avec SELECT au lieu de EXAMINE*. Par contre, les Flags ne changeront pas si l'accès à la boite été fait par EXAMINE.