Réseaux informatiques

Stéphane Bortzmeyer
<stephane@bortzmeyer.org>

27 février 2008

Un petit programme tout seul

```
import urllib2
import time
results_filename = "test.out"
url = "http://www.pasteur.fr/"
n = 100 # Read only the first n characters

results_file = open(results_filename, 'w')
try:
    web_data = urllib2.urlopen(url).read(n)
except Exception:
    web_data = None
results_file.write("Result for \"%s\" at %s is \"%s\"\n" % \
                   (url, time.strftime("%H:%M:%S",
                                       time.localtime(time.time(
                    web_data))
results_file.close()
```

Que fait ce programme ?

1. Il ouvre en écriture un fichier nommé "test.out" (`open()` est une fonction pré-définie en Python),
2. Il récupère une page Web, "http ://www.pasteur.fr/" (le module urllib2 fait partie de la bibliothèque standard),
3. Il stocke les 100 premiers caractères de cette page dans le fichier (`write()` est une méthode des objets Fichier).

Les réseaux

Aujourd'hui, il n'y a presque plus de programmes sans accès au réseau.

Internet a bouleversé la biologie. . . comme beaucoup d'autres secteurs d'activité.

Mais comment marchent les réseaux ?

Que se passe t-il derrière `urllib2.urlopen(url).read(n)` ?

Où on retrouve le modèle en couches

On analyse en général les réseaux avec cinq, sept ou neuf couches. Ici, on en mettra six :

- La couche application : ce que voit l'utilisateur.
- La (demi-)couche d'infrastructure : les applications qui n'ont pas d'utilité pour elles-mêmes comme le DNS,
- La couche transport : réémission des paquets perdus, et rangement des paquets dans l'ordre,

Les couches basses

- La couche réseau : première couche à aller d'un bout à l'autre du réseau, l'acheminement des paquets de données et le routage,
- La couche liaison : l'accès au câble,
- La couche physique : les câbles et les signaux électriques,

Ce que fait chaque couche

Elle rend des services à la couche supérieure

Et, pour cela, elle se sert de la couche inférieure

Cela ressemble au modèle que nous avons utilisé pour le SE.

Les protocoles

Un protocole est un **langage** que doivent parler les mises en œuvre de chaque couche, pour se faire comprendre du partenaire.

HTTP est un protocole de couche Application.

TCP est un protocole de couche Transport.

IP est un protocole de couche Réseau.

Ensemble, TCP et IP forment la base de l'Internet. On les appelle souvent simplement « TCP/IP ».

Les couches, de haut en bas

Revenons à notre `urllib2.urlopen(url).read(n)`.

D'abord, Python doit parler un protocole Application, ici HTTP (le `http :` de l'URL).

HTTP, protocole principal du Web, est décrit dans le RFC 2616. (RFC = textes sacrés de l'Internet.)

C'est un protocole **client-serveur**, un serveur attend des requêtes, un client vient demander (comme au café).

Les couches, suite

Il y a bien d'autres protocoles Application : SMTP (courrier), XMPP (messagerie instantanée), SSH (connexion à distance), e2DK (transfert de fichiers pair-à-pair, dans le réseau eDonkey), . . .

Des logiciels comme tcpdump ou wireshark sont très pratiques pour observer une session et l'analyser. L'extension Firefox "Live HTTP headers" permet d'observer HTTP depuis son navigateur.

Examiner ce qui se passe

Mais on peut aussi utiliser un logiciel client HTTP avec un mode bavard :

```
% curl -v http://www.pasteur.fr/
...
> GET / HTTP/1.1
> User-Agent: curl/7.16.4 (i486-pc-linux-gnu) libcurl/7.16.4 Ope
> Host: www.pasteur.fr
> Accept: */*
>
< HTTP/1.1 301 Moved Permanently
< Server: Apache/1.3.31 (Unix) mod_ssl/2.8.17 OpenSSL/0.9.7b mod
< Location: http://www.pasteur.fr/ip/
< Content-Type: text/html; charset=iso-8859-1
```

Explications du mode bavard

Le $>$ indique les requêtes du client au serveur, le $<$ les réponses du serveur. Le protocole, c'est décider que, pour demander un fichier, on envoie la chaîne de caractères GET et que le serveur répond par un code à trois chiffres (ici 301).

La couche d'infrastructure

Il existe un certain nombre de protocoles qui sont officiellement dans la couche Application mais qui ne rendent pas un service directement perceptible par l'utilisateur. Par exemple :

- DNS
- DHCP (attribution d'adresses IP)
- NTP (synchronisation d'horloges)

DNS, Domain Name System

Le DNS est l'annuaire de l'Internet.

Il permet de trouver les adresses, qu'utilisera la couche Réseau à partir d'un nom.

Et bien d'autres choses.

Les **noms de domaines** sont écrits sous la forme de labels séparés par des points, par exemple `munster3.sis.pasteur.fr`.

DNS, Domain Name System

Le DNS est l'annuaire de l'Internet.

Il permet de trouver les adresses, qu'utilisera la couche Réseau à partir d'un nom.

Et bien d'autres choses.

Les **noms de domaines** sont écrits sous la forme de labels séparés par des points, par exemple `munster3.sis.pasteur.fr`.

Le label le plus à droite est nommé le TLD, *Top-Level Domain*. "`fr`" représente la France, "`cn`" la Chine, "`ci`", la Côte d'Ivoire, etc.

Pourquoi le DNS

Le DNS fournit un système d'identificateurs relativement stables (alors que les adresses sont plus volatiles).

C'est à partir du DNS qu'on construit les **URL**, les adresses du Web comme `http://www.pasteur.fr/recherche/RAR/RAR2006/Bive.html`.

La notion de paquet

Les bits ne voyagent pas en solitaire sur le câble. Ils sont regroupés en **paquets**. Un paquet a un **en-tête** qui contient les **méta-données** et des **données**.

La couche Transport

Son rôle principal est de rattraper les paquets perdus par les couches inférieures (réclamer la retransmission, les remettre dans l'ordre).

Elle présente un service de flux de données fiable, même si le réseau sous-jacent ne l'est pas. Une application peut donc lire et écrire (comme avec notre `read()`) sur une connexion TCP, comme elle le ferait sur un fichier.

Le protocole de transport le plus connu est TCP (Transmission Control Protocol).

La couche Réseau

Son rôle principal est l'acheminement des données d'un bout à l'autre de l'Internet. C'est la dernière couche à fonctionner de bout en bout.

Elle définit le format des paquets IP (*Internet Protocol*) et le mécanisme d'**adressage**.

Exemple, avec la version 6 du protocole IP : les adresses ont 128 bits et sont écrites comme huit groupes de quatre chiffres hexadécimaux comme `2001:660:AF:42:345:BEEF:A0:1`.

Routage

Ces adresses IP permettent aux **routeurs** d'acheminer les données jusqu'à destination.

Pour connaitre les routes, les routeurs exécutent en permanence entre eux des protocoles de routage comme BGP ou OSPF. Par exemple, l'attaque pakistanaise contre YouTube ce week-end était une attaque contre BGP.

Petite pause avec tcpdump

```
09:50:54.501309 IP 192.134.4.11.44320 > 217.70.190.232.25: \
        S 4098014021:4098014021(0) win 5840 \
        <mss 1460,sackOK,timestamp 1019979251 0,nop,wscale 7>
09:50:54.501338 IP 217.70.190.232.25 > 192.134.4.11.44320: \
        S 2310063626:2310063626(0) ack 4098014022 win 5792 \
        <mss 1460,sackOK,timestamp 60396256 1019979251,nop,wsca
```

On voit ici l'échange de paquets IPv4 entre deux machines. De gauche à droite, l'heure, l'**adresse IP** de l'expéditeur et le **port** utilisé (25 = SMTP, le courrier), celle du destinataire, et un décodage des données, indiquant le protocole de transport, ici TCP.

La couche Liaison

Son rôle est de gérer l'accès au support physique. Par exemple, en WiFi, une seule station peut transmettre sur une fréquence donnée. Un protocole permet de s'assurer que c'est bien le cas.

Notez l'analogie avec le protocole qu'on utilise au téléphone ou à une table de restaurant pour s'assurer qu'une seule personne parle à la fois.

La couche Physique

Nous arrivons ici dans le domaine des physiciens et des électroniciens. Ici, on parle en hertz et en ohms, on manipule des objets physiques et on se salit les mains.

Exemple de couche physique

Les communications intercontinentales passent en général par des câbles sous-marins. Ceux-ci sont des fibres optiques posées sur le fond de la mer. En haute mer, la profondeur les met à l'abri. Mais le câble doit un jour **atterrir**. Remontant dans les eaux peu profondes, il est alors très vulnérable.

Exemple de couche physique

Les communications intercontinentales passent en général par des câbles sous-marins. Ceux-ci sont des fibres optiques posées sur le fond de la mer. En haute mer, la profondeur les met à l'abri. Mais le câble doit un jour **atterrir**. Remontant dans les eaux peu profondes, il est alors très vulnérable.

En janvier 2007, une série noire au Moyen-Orient a coupé de nombreux câbles, menant certains à affirmer qu'une guerre de basse intensité était en cours.

James Bond et le câble sous-marin, 2

En fait, la quasi-totalité des coupures sont dues à des bateaux de pêche. Filets et ancres sont plus dangereux que les bombes.

James Bond et le câble sous-marin, 2

En fait, la quasi-totalité des coupures sont dues à des bateaux de pêche. Filets et ancres sont plus dangereux que les bombes.

La coupure nécessite de renvoyer sur le site un bateau de réparation. Les gens de la couche Physique travaillent au grand air, par tous les temps, eux !

wireshark et le modèle en couches

Wireshark est très pratique pour explorer interactivement les différentes couches.

[Démonstration interactive avec wireshark et des traces sauvegardées.]

Les couches et le déboguage

Le modèle en couches est très pratique pour structurer le déboguage.

Par exemple, un utilisateur crie « Ça marche pas ». Après une longue psychanalyse, on arrive à en extraire un rapport plus précis « Pas moyen de se connecter à www.youtube.com ». On va alors pouvoir procéder scientifiquement (l'informatique est une science expérimentale).

Étapes du déboguage

1. On regarde avec un navigateur Web et on ne voit pas sa vidéo favorite. On arrête avec le navigateur, ce n'est pas un bon outil de déboguage, les messages d'erreur d'Internet Explorer ne valent rien.
2. On teste avec un outil plus adapté, par exemple curl.
    1. *Couldn't resolve host* : nom inexistant, le problème est quelque part dans le DNS
    2. *couldn't connect to host* : le problème est dans la couche Réseau, il faut passer à ping et traceroute

L'aventure d'un urlopen()

1. Le programmeur a écrit `urlopen(url).read(n)`. Python l'exécute.
2. L'interpréteur Python appelle la fonction `urlopen` de la bibliothèque standard.
3. Après beaucoup de code Python (la fonction est très souple et peut faire beaucoup de chose), on arrivera (dans le module `httplib`) à la fonction `socket.getaddrinfo` de traduction de nom en adresse. Elle appelle à son tour divers appels systèmes pour finir par une requête DNS (on ne détaille pas).

urlopen(), suite

1. On a alors l'adresse IP. Python appelle `self.sock.connect` qui ouvre une connexion TCP avec le serveur. Le premier paquet sera la demande d'ouverture de la connexion.
2. TCP est entièrement géré par le noyau Unix. Celui-ci fabrique le paquet, détermine l'interface réseau de sortie et transmet le paquet au **pilote** Ethernet. Celui-ci suit le protocole Ethernet d'accès au câble et finit par envoyer les signaux physiques sur le câble.

urlopen(), suite et fin

1. Reçu par le premier routeur, le paquet est lu et renvoyé sur l'interface de sortie. Et ainsi de suite à chaque routeur.
2. Le paquet arrive au serveur, dont le noyau gère l'établissement de la connexion TCP, avant de prévenir l'application (probablement le serveur HTTP Apache).
3. La connexion TCP est désormais établie. Python va pouvoir exécuter `read` pour faire passer les données.