**Ecole Polytechnique de l'Université François Rabelais de Tours**
Département Informatique
64 avenue Jean Portalis
37200 Tours, France
Tél. +33 (0)2 47 36 14 14
polytech.univ-tours.fr

## Projet Architecture, Système et Réseaux

## 2018-2019

# Crawling web et requête HTTP par serveur proxy

**Tuteur académique**
Mathieu Delalandre

**Étudiants**
Thomas Couchoud (DI5)
Victor Coleau (DI5)

12 octobre 2018

# Liste des intervenants

| Nom | Email | Qualité |
|---|---|---|
| Thomas Couchoud | thomas.couchoud@etu.univ-tours.fr | Étudiant DI5 |
| Victor Coleau | victor.coleau@etu.univ-tours.fr | Étudiant DI5 |
| Mathieu Delalandre | mathieu.delalandre@univ-tours.fr | Tuteur académique, Département Informatique |

# Avertissement

Ce document a été rédigé par Thomas Couchoud et Victor Coleau susnommés les auteurs.

L'Ecole Polytechnique de l'Université François Rabelais de Tours est représentée par Mathieu Delalandre susnommé le tuteur académique.

Par l'utilisation de ce modèle de document, l'ensemble des intervenants du projet acceptent les conditions définies ci-après.

Les auteurs reconnaissent assumer l'entière responsabilité du contenu du document ainsi que toutes suites judiciaires qui pourraient en découler du fait du non respect des lois ou des droits d'auteur.

Les auteurs attestent que les propos du document sont sincères et assument l'entière responsabilité de la véracité des propos.

Les auteurs attestent ne pas s'approprier le travail d'autrui et que le document ne contient aucun plagiat.

Les auteurs attestent que le document ne contient aucun propos diffamatoire ou condamnable devant la loi.

Les auteurs reconnaissent qu'ils ne peuvent diffuser ce document en partie ou en intégralité sous quelque forme que ce soit sans l'accord préalable du tuteur académique et de l'entreprise.

Les auteurs autorisent l'école polytechnique de l'université François Rabelais de Tours à diffuser tout ou partie de ce document, sous quelque forme que ce soit, y compris après transformation en citant la source. Cette diffusion devra se faire gracieusement et être accompagnée du présent avertissement.

# Pour citer ce document

Thomas Couchoud et Victor Coleau, *Crawling web et requête HTTP par serveur proxy*, Projet Architecture, Système et Réseaux, Ecole Polytechnique de l'Université François Rabelais de Tours, Tours, France, 2018-2019.

```
@mastersthesis{
    author={Couchoud, Thomas and Coleau, Victor},
    title={Crawling web et requête HTTP par serveur proxy},
    type={Projet Architecture, Système et Réseaux},
    school={Ecole Polytechnique de l'Université François Rabelais de Tours},
    address={Tours, France},
    year={2018-2019}
}
```

Document mis en forme sous LaTeX
grâce à la classe de document `polytech.cls` V2.2.5 (2017/05/02)

# Table des matières

# Table des figures

**Liste des tableaux**



**3 Les proxy**

# Liste des tableaux

# Première partie

# Introduction

Dans notre société moderne l'Internet occupe une place très importante. Il offre une quantité pharaonique d'information en libre accès. Parmi ces sources d'information, on trouve notamment des « wikis » qui sont des encyclopédies collaboratives permettant la large diffusion de données.

Malgré leur apparente générosité et leur connaissance des pratiques, ces sites n'apprécient que peu que les données qu'ils fournissent en soient extraites.

Cette nouvelle mane d'information attire les convoitises et polarise les comportements. D'un côté nous retrouvons les collecteurs de données cherchant à en agréger et stocker de plus en plus de leur propre chef. De l'autre les sites mettant à disposition l'information dont le but paradoxal est de fournir gratuitement tout en conservant l'exclusivité.

Cela entraine une guerre technologique entre crawlers et sites web. Les premiers développent des technologies de plus en plus efficaces, rapides et discrètes. Les seconds cherchent à contrecarrer les premiers grâce à des techniques de détection de plus en plus sophistiquées.

Le but de ce projet est d'étudier à la fois les techniques mises en place par les crawler pour se rendre invisible et celles mise en place par les sites pour se défendre. Cette recherche se concrétisera par la réalisation d'un crawler effectuant ses connexions au travers d'un proxy.

**Figure 1** – *Crawler*

Deuxième partie

# Veille technique

# 1 Stratégies de défense

Afin de se défendre face aux demandes massives que peuvent représenter les crawlers, les créateurs de sites web ont imaginés plusieurs méthodes de contre-attaque. Dans cette partie nous allons en développer quelques unes.

## 1 Liste noire

Le principe de base d'une liste noire est de bannir du site les « utilisateurs » trop agressifs. Un « utilisateur » peut être un compte inscrit sur le site ou plus simplement une adresse IP requêtant le serveur.

La problématique principale de cette méthode est différencier un utilisateur humain d'un automate. Afin de prendre la décision de bannir ou non, plusieurs moyens sont à disposition des administrateurs :

- User-agent : Le user-agent (UA) est un champ renseigné dans l'entête d'une requête HTTP. Son but est d'identifier l'outil qui a engendré cette demande. Par exemple un UA de Firefox ressemble à « Mozilla/5.0 (X11 ; Ubuntu ; Linux_x86_64 ; rv :62.0) Gecko/20100101 Firefox/62.0 » tandis que celui d'un bot Google est de la forme « Mozilla/5.0 (compatible ; Googlebot/2.1 ; +http ://www.google.com/bot.html) ».

  Grace à cette identification, il est possible de filtrer les requêtes pour n'accepter que celles provenant des navigateurs web standards.

- Adresse IP : Une approche peut être de se baser sur l'adresse IP permettant d'identifier une machine ou un réseau précis.

- Comportement de l'utilisateur : A partir des méthodes d'identifications précédentes, il est possible de définir une stratégie de bannissement plus juste. En effet bannir tous les navigateurs qui ne sont pas Internet Explorer ou bien toutes les adresses IP commençant par 1 n'est que peu pertinent.

  Il est alors possible d'étudier les méthodes d'explorations du site afin de dénicher les comportement indésirables. Ces derniers peuvent être plus ou moins simples à observer.

  – Un nombre de requêtes humainement impossible (ex : 5 requêtes par seconde).

  – Des temps de visite des pages très courts (ex : 0.5 secondes par page).

– Une fréquence de requêtage régulière (ex : toutes les secondes).

– Les sauts de pages à d'autres non reliées par hyperlien.

La pertinence de cette méthode de bannissement dépends grandement de la qualité de reconnaissance des automates. En effet, il se peut qu'un automate imite parfaitement le comportement humain, et donc ne soit pas détecté, tout comme un comportement humain inhabituel pourrait être banni par erreur. L'objectif est donc de développer des méthodes de reconnaissance adaptées afin de ne pas perdre des visiteurs désirés.

## 2 CAPTCHA

Le CAPTCHA est une famille de tests de Turing ayant pour but de différencier un utilisateur humain d'un automate. Plusieurs types de CPATCHAs sont imaginables :

- Reconnaitre une site de lettres altérées (ex : déformées, barrées, avec des trous, ...).
- Cocher une case qui vérifie le comportement précédent de l'utilisateur.
- Reconnaitre des parties spécifiques d'une image.



## 3 Modification du DOM

Une autre approche serait non plus d'essayer de rejeter les automates mais de leur compliquer la tâche.

Une méthode possible consiste à modifier constamment et régulièrement la structure du site (nom des classes, IDs, ...). De même, il est envisageable de remplacer certaines parties du site (notamment le texte) par des images.

De ce fait les automates spécifiques à un site donné devront être réadaptés fréquemment ce qui peut décourager leur développeur.

## 4 Attrapes-bots

Afin de distinguer un humain d'un bot, il serait envisageable de baser notre différenciation sur l'exploration ou non d'une page web normalement non-visible d'un utilisateur lambda. Html

/ CSS offrent la possibilité de masquer certains éléments tout en les incluant au sein du code source de la page.

Cependant, un crawler innatentif ne fera pas la différence entre des éléments visibles ou non. On pourrait donc renforcer la sécurité du site grâce à des liens pigés et bannir toute IP s'y rendant.

# 2 Stratégies de masquage

Comme décris précédemment, les gestionnaires de sites web tentent constamment de bloquer l'accès à leur contenu au travers de diverses méthodes. Ces dernières reposent principalement sur le bloquage d'un identifiant unique à la personne suite au repérage d'un comportement suspect.

Les personnes attaquant les sites ont donc pour but de contourner ces mesures, soit en ayant plusieurs identifiants à leur disposition soit en essayant d'adopter un comportement proche d'un utilisateur normal.

## 1 Comportement du crawler

Une première mesure qui pourrait être envisagée afin d'éviter que le crawler soit identifié est de mofidifer les headers de requête HTTP par défaut mis en place par les librairies permettant d'effectuer des demandes HTTP. En effet, si l'on prend par exemple la librairie urllib de Python, on remarque que le UA par défaut est « Python-urllib/3.4 » et que le Accept-Encoding est « identity ». Avec un tel UA, le modérateur peut facilement comprendre qu'un programme Python est à l'origine de ces requêtes et non un utilisateur humain utilisant un navigateur web standard. Il serait alors plus judicieux de remplacer notre UA par celui d'un navigateur web répendu (Chrome, Firefox, Edge, etc.).

De même, le Accept-Encoding peut, dans certains cas, compromettre l'anonymat du crawler. Cependant, la modification de certaines informations du header peut entrainer des changements sur le site en question. Par exemple, le header Accept-Langage indique les préférences utilisateur concernant la langue. Il se peut donc qu'un site mette à disposition deux versions en fonction de la langue souhaitée.

Une deuxième méthode serait de prendre en charge les cookies au travers de JavaScript. En effet, ceux-ci peuvent être utilisés afin de reconnaitre un utilisateur ayant visité le site peu de temps auparavant. Si un crawler se rend sur le site de manière intensive sans présenter ce cookie, le serveur pourrait se rendre compte qu'il ne s'agit pas d'un utilisateur faisant des requêtes succesives.

Cepndant, enregistrer les cookies, peut aussi jouer en notre défaveur : ceux-ci sont un façon de garder un trace d'un utilisateur précis et donc de l'identifier (Voir Section 1 (Chapitre 1)).

Enfin, un trop grand nombre de requêtes en très peu de temps trahit un comportement robotique. Il est donc important de controler la vitesse de notre crawler afin de ne pas surcharger la bande passante du site et risquer de se faire bannir.

Cette recommandation est en opposition avec les techniques de parrallèlisation souvent mises en place afin d'accélérer la vitesse et l'efficacité du crawler.

## 2 Utilisation de plusieurs identifiants

Comme vu précédemment, les principales techniques anti-crawler reposent sur le bannissement d'adresse IP irrespectueuses. Une méthode de contournement basique est donc de changer régulièrement d'adresse IP. Les proxys sont donc une alternative parfaite (Voir Chapitre 3).

De plus, certains sites utilisent des sessions ou comptes utilisateurs. Un crawler pourrait donc en tirer parti en s'indentifiant officiellement sur le site afin de se faire passer pour un utilisateur standard.

Il est d'ailleurs poissible de créer et d'utiliser plusieurs comptes en parrallèle afin de répartir la charge de requêtes.

# 3 Les proxy

## 1 Késako ?

Un proxy est un composant logiciel servant d'intermédiaire entre deux hôtes afin de faciliter ou de surveiller leurs échanges.

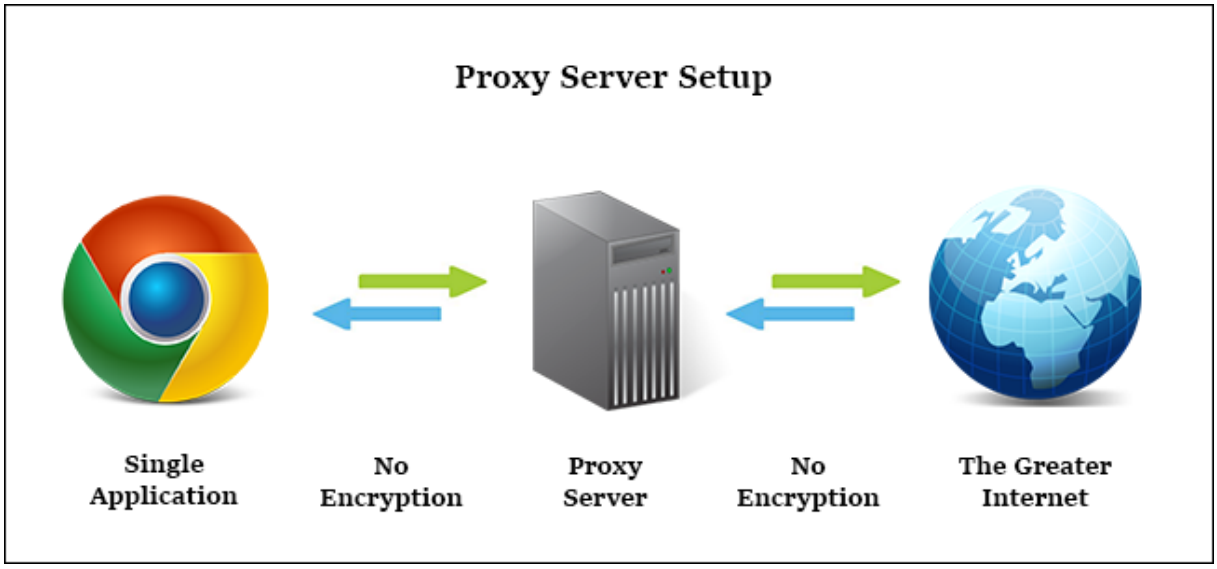


**Figure 1** – *La place d'un proxy dans une communicatiob*

### 1.1 Avantages

Les proxys peuvent par exemple servir à contourner certains filtrages. Supposons le cas d'un pays qui bloque l'accès à certains sites, en se connectant à un proxy non bloqué, l'utilisateur pourra accéder à son site au travers de ce dernier car le proxy ne dispose pas des mêmes règles de filtrage.

A l'inverse, certains établissements scolaires ou entreprises limitent l'accès à certains sites grace à un serveur proxy. En effet, toutes les requêtes effectuées par les utilisateurs du réseau passe par ce serveur intermédiaire qui bloque les sites dont l'adresse a été spécifiquement interdite.

Un autre avantage de l'utilisation d'un proxy est de pouvoir surfer anonymement. Les sites visités n'ont conscience que de l'adresse du proxy et non de(s) utilisateur(s) caché(s) derrière.

De plus un proxy permet le masquage de son lieu de connexion. En effet le proxy peut ne pas être situé dans le même pays que l'utilisateur. Si un site se base sur un système de géolocalisation pour afficher son contenu (YouTube, Google, Google Maps, etc.), sera prise en compte la géolocalisation du proxy.

### 1.2 Inconvénients

Bien que les poxys offrent de nombreux avantages, ils ont aussi certains inconvénients. S'agissant d'une plateforme reliant un utilisateur au web et effectuant les requêtes du premier, celui-ci voit passer tous les échanges. Certains proxys pourraient alors les enregistrer à des fins malveillantes.

L'intérêt d'un proxy étant important, il centralise toutes les requêtes d'une structure et peut donc être saturé ralentissant par la même occasion la connexion de tous les utilisateurs. De manière générale on peut dire qu'une connexion internet passant par un proxy sera toujours plus lente qu'une connexion directe.

## 2 Alternative - VPN

VPN est un acronyme signifiant « Virtual Private Network ». Tout comme les proxys ils permettent de faire apparaître notre navigation internet comme provenant d'une adresse IP distante.

A la différence d'un proxy qui se configure par application au cas par cas (par exemple dans Firefox on peut avoir une configuration différente de celle dans Chrome). En revanche un VPN capture l'ensemble des échanges réseau de la machine et est configuré directement dans le système d'exploitation. Les différentes applications de la machine n'ont donc pas conscience de cette subtilité.

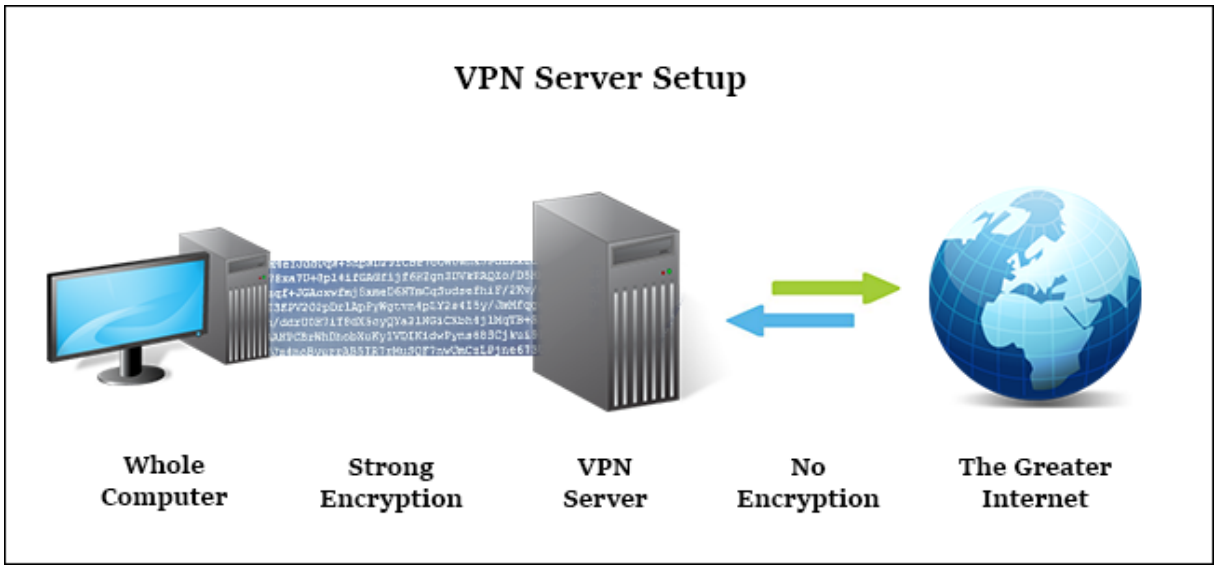


**Figure 2** – *La place d'un VPN dans une communication*

Dans le cas de notre crawler, il parait plus adapté d'utiliser un proxy. En effet seul ce dernier requiert d'être « masqué » et passer toute notre connexion au travers d'un VPN semble être un peu overkill.

## 3 Présentation de proxys

| Nom | Prix | Pays | Avantages | Inconvénients |
|---|---|---|---|---|
| Proxy6 | 1.25$ par IPv4 par mois | Un choix très large | API pour développeurs & Le moins cher | |
| BuyProxies | 2$ par serveur proxy par mois | Un choix très large | Bande passante illimité & renouvellement mensuel des proxys | |
| FoxyProxy | 10$ par IP par mois | - | | Pays inconnu & Le plus cher |
| ProxyRack | 40$ pour 50 connexions simultanées par mois | Un choix moyen de pays | Rotation des adresses IP & Bande passante illimitée | Premier pack cher |

**Table 2** – *Caractéristiques utilisateurs*

Troisième partie

# Application : Réalisation d'une crawler avec proxy

# 4 Crawler basique

Dans le cadre de ce projet nous devons tester les stratégies de masquage adoptables par un crawler. Afin de maitriser au mieux ces tests, nous avons souhaité développer nous même un tel logiciel.

Afin d'obtenir un résultat de crawling pertinent tout en restant simple d'implémentation, nous avons choisi de se concentrer sur la recherche et téléchargement des images. Pour ce faire, notre crawler explore tous les liens d'une page ainsi que toutes les images. Il téléchargera les images trouvées tandis qu'il continuera d'explorer les liens d'un même domaine.

Dans l'optique de garder le crawler simple, le javascript n'est pas supporté et seuls les liens dans les balises « a » et « img » sont traités.

## 1 Méthode implémentée

Dans un premier temps, les problématiques de bloquage ne seront pas prises en compte, seule l'efficacité prime. Pour cela, nous avons pensé à un système supportant le multi-threading augmentant la capacité d'acquisition des données.

Le programme a été découpé en deux parties : les crawlers et les downloaders.

Les crawlers sont en charge de l'exploration du site web et remplissent deux queues. La première contient les prochains liens à explorer et la seconde les liens des images à télécharger.

Les downloaders, eux, ne font que lire les liens de la queue qui leur est associée et télécharge les images.

L'utilisation de ces queues nous permet de séparer l'exploration et le téléchargement dans des threads différents mais aussi d'avoir plusieurs crawlers et plusieurs downloaders.

### 1.1 Crawlers

Le travail d'un crawler se décompose de la manière suivante :

- Acquisition d'un lien depuis la queue des liens. Si aucun lien n'est disponible, le thread s'endort temporairement et reprendra au début.
- Ajout de ce lien à la liste des liens déjà explorés.

- Téléchargement de la page HTML.
- Lecture de cette page et récupération des différents liens des balises « a » et « img ».
- Tri des liens obtenus :
  - Si le lien correspond à une image (basé sur l'extension) : on vérifie que cette image n'a pas déjà été téléchargée. Si tel est le cas, on l'ajoute dans la queue des images (queue sans doublons).
  - Sinon : il vérifie si le lien a déjà été exploré. Si ce n'est pas le cas on vérifie qu'il fait partie du même domaine que le lien en cours d'exploration. Si tel est le cas on l'ajoute dans la queue des liens à explorer (queue sans doublons).

## 1.2 Downloaders

Le travail d'un downloader se décompose de la manière suivante :

- Acquisition d'un lien depuis la queue des images. Si aucun lien n'est disponible, le thread s'endort temporairement et reprendra au début.
- Ajout de ce lien à la liste des images déjà téléchargées.
- Récupération du nom de l'image à partir du lien.
- Acquisition du contenu :
  - Si le ficher de sortie est déjà présent, on ne fait que renseigner le lien dans un ficher texte au nom de l'image (contiendra tous les liens menant potentiellement à cette dernière).
  - Sinon, on télécharge l'image.

# 2 Résultats de tests

| Site | Observations |
|---|---|
| Wikipedia | |
| Qwertee | |
| Etam | |
| Google | |
| 4chan | |
| Tinder | |
| PrettyLittleThings | |
| LaDechetterieDuWeb | |

**Table 2** – *Résultats sur différents sites*

# 5 Utilisation de stratégies de masquage

# 6 Proxy

Quatrième partie

# Conclusion

# Annexes

# A Acronymes

**UA** user-agent. 3, 6

# Crawling web et requête HTTP par serveur proxy

**Résumé**

**Mots-clés**

**Abstract**

**Keywords**

**Tuteur académique**
Mathieu Delalandre

**Étudiants**
Thomas Couchoud (DI5)
Victor Coleau (DI5)