HTTP

Damien Gros

19 octobre 2023

Plan du cours

Préambule

Versions HTTP
- HTTP v0.9
- HTTP v1.0
- HTTP v1.1

Les fonctionnalités d'HTTP
- Les cookies
- HTTP et la sécurité
- Authentification
- Un peu d'admin..
- Autour d'HTTP

Plan du cours

Préambule

Versions HTTP
- HTTP v0.9
- HTTP v1.0
- HTTP v1.1

Les fonctionnalités d'HTTP
- Les cookies
- HTTP et la sécurité
- Authentification
- Un peu d'admin..
- Autour d'HTTP

Hyper Text Transfert Protocol

- Protocole de la couche application ;
- Fondement du World Wide Web (www) ;
- HTTP : créé en 1990 par *Berners-Lee*
- Différentes versions :
  - HTTP v0.9 : non normalisé, 1990 ;
  - HTTP v1.0 : RFCc1945, en 1996
  - HTTP v1.1 : RFC2068, en 1997
  - HTTP v2.0 : en cours de rédaction par IETF (The Internet Engineering Task Force)
- NB : Request For Comments : définition des standards.

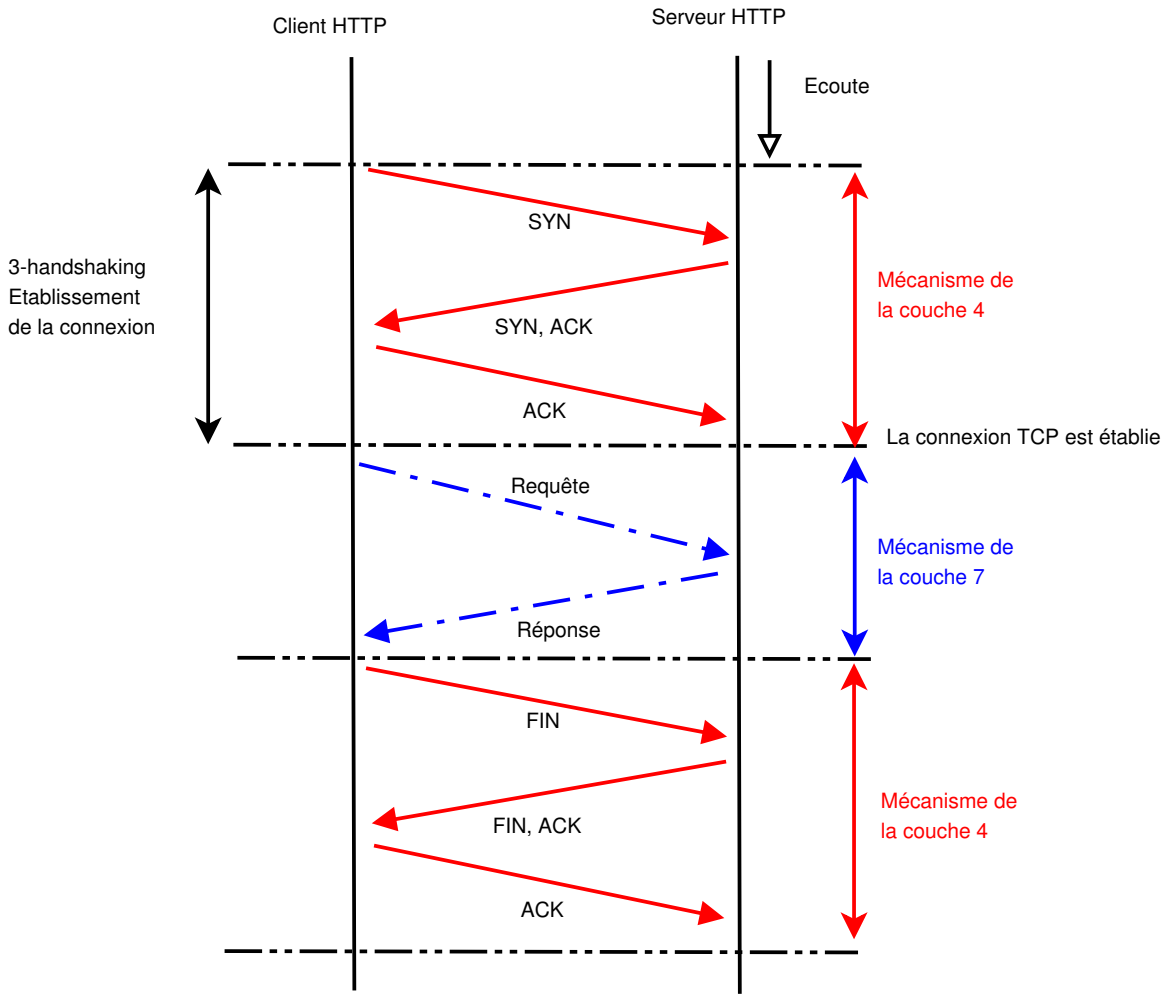
HTTP

- Protocole simple :
  - Établissement d'une connexion TCP entre un client et un serveur ;
  - Le client envoie une requête au serveur (demande de document) ;
  - Le serveur envoie la réponse au client (envoie du document) ;
  - Le serveur ferme la connexion.
- Objectif : échanger des ressources entre un client et un serveur.
  - Protocole de rapatriement de documents ;
  - Protocole de soumission de formulaires.

Couches inférieures

- HTTP est un protocole de la couche application (7) ;
- *Il n'a pas conscience des protocoles utilisés en dessous* ;
- On peut donc faire de l'HTTP sur TCP et UDP ;
- Dans la pratique, on le fait sur TCP (on le verra comme ça tout au long de ce cours).

Communication HTTP basique



Un peu de vocabulaire

- Terminologies :
  - HTTP : Hyper Text Transport Protocole ;
  - URL : Uniform Ressource Locator ;
  - URI : Uniform Ressource Identifier ;
  - HTML : Hyper Text Markup Language ;

Différences entre URI et URL ?

(extrait des RFC) :

- URI is a compact sequence of characters that identifies an abstract or physical resource ;
- URL refers to the subset of URIs that, in addition to identifying a resource, provide a means of locating the resource by describing its primary access mechanism (e.g., its network “location”).
- www.esiea.fr : URI
- http ://www.esiea.fr : URL
- ftp ://www.esiea.fr : URL
- ldap ://www.esiea.fr : URL

HTTP

- Les éléments du HTTP
  - Serveurs : éléments qui se *contentent* de répondre à un client ;
    - IIS, Apache, Light httpd, Zeus Web Server, nginx, etc.
  - Clients : plus complexes que les serveurs, présentent les données envoyées par le serveur à l'utilisateur ;
    - "Classiques" : IE, Opéra, Firefox, Chrome, Safari, SeaMonkey, etc.
    - "Textuels" : lynx, elinks, curl, links, etc
    - "En ligne de commande" : wget ;
    - Aspirateurs de site ;
    - Robots d'indexation ;
    - Tout programme capable de traverser un proxy HTTP.

Rôle du serveur HTTP

- Transformation de l'URL en fichier ou en script ;
- Vérification d'identité
  - Le client est-il celui qu'il prétend être ?
- Vérification d'accès
  - Le client est-il autorisé à effectuer cette requête ?
  - ACL, etc.
- Constitution de l'entête de la réponse
  - Type MIME des données (détaillé par la suite)
  - mime.types fichier de correspondance extension vers type MIME
- Taille des données, Langage, etc.
- Envoi de la réponse au client
  - éventuellement transformé à la volée
- Mise à jour des journaux d'audit (log) access_log, error_log, ...

Types de requête

- GET : demande une ressource, la plus courante, ne modifie pas la ressource ;
- HEAD : demande des informations sur la ressource, mais ne demande pas la ressource ;
- POST : pour ajouter une ressource ;
- OPTIONS : obtention des informations sur les options de connexion ;
- CONNECT : pour utiliser un proxy comme tunnel de connexion ;
- TRACE : pour diagnostiquer la communication ;
- PUT : ajoute/remplace une ressource sur le serveur ;
- DELETE : supprimer une ressource ;

Plan du cours

Préambule

Versions HTTP
- HTTP v0.9
- HTTP v1.0
- HTTP v1.1

Les fonctionnalités d'HTTP
- Les cookies
- HTTP et la sécurité
- Authentification
- Un peu d'admin..
- Autour d'HTTP

HTTP V0.9

1. Connexion du client HTTP au serveur HTTP ;
2. Utilisation de la méthode GET ;
3. Réponse du serveur HTTP ;
4. Le serveur ferme la connexion HTTP $\rightarrow$ fin de la connexion.

Une connexion par fichier !

HTTP V0.9

Illustration avec un serveur local et telnet

HTTP V0.9

- On ne spécifie pas la version du protocole utilisé (le serveur la reconnait) ;
- Le serveur répond directement !

Requête du client HTTP

```
GET /
```

Réponse du serveur HTTP

```
<html>
[..]
</html>
```

HTTP V1.0

- Utilisation des entêtes, des MIME ;
- MIME :
  - *Multi-Purpose Internet Mail Extensions* ;
  - Définit le type de média que le serveur envoie ;
- Même gestion des communications qu'en HTTP/0.9
  1. Connexion du client HTTP au serveur HTTP ;
  2. Utilisation de la méthode GET ;
  3. Réponse du serveur HTTP : **envoie de l'entête** puis du reste (contenu du fichier) séparé par une ligne vide ;
  4. Le serveur ferme la connexion HTTP $\rightarrow$ fin de la connexion.

HTTP V1.0

Illustration avec un serveur local et telnet.

HTTP V1.0

- On spécifie la version HTTP utilisé ;
- On laisse une ligne vide ;
- Le serveur envoie une entête puis le corps du message.

HTTP V1.0

Requête envoyée par le client HTTP :

```
GET / HTTP/1.0
Host : foo.com
Referer : http ://foo.com
User-Agent : Firefox 3.5
< ligne vide >
```

- Type de requête avec la version du protocole utilisé ;
- Host : précise le nom du serveur quand il y a plusieurs serveurs web sur la même IP (hébergement mutualisé) ;
- Referer : d'où vient le visiteur ;
- User-Agent : nom du client HTTP (version et parfois OS).

HTTP V1.0

Réponse envoyé par le serveur

```
HTTP/1.1 403 Forbidden
Date: Tue, 02 Sep 2014 08:36:56 GMT
Server: Apache/2.4.6 (CentOS)
Last-Modified: Tue, 17 Jun 2014 16:00:47 GMT
Accept-Ranges: bytes
Content-Length: 4880
Content-Type: text/html; charset=UTF-8

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.1//EN" "http://www.w3.org/TR/
    xhtml11/DTD/xhtml11.dtd"><html><head>
<meta http-equiv="content-type" content="text/html; charset=UTF-8">
```

Structure d'une requête

- GET / HTTP/1.0
- GET : méthode de communication avec le serveur : je désire un fichier ;
- / : je veux le truc à la racine du serveur HTTP ;
- /index.html : je veux le fichier index.html à la racine du serveur HTTP ;
- /toto/index.html : je veux le fichier index.html qui se trouve dans le dossier toto
- HTTP/1.0 : je fais cette requête en utilisant le protocole HTTP version 1.0

HTTP V1.0

- Version du protocole utilisé, statut HTTP et signification textuelle ;
- Date : quand la page a été générée ;
- Server : Nom du programme faisant office de serveur, version, OS ;
- Last-Modified : dernière modification de la page ;
- Content-Type : type MIME de la ressource ;
- Content-Length : taille de la ressource (en octet).

Statut HTTP

- 1XX : informatif, non utilisé à ce jour ;
- 2XX : Succès de l'opération :
  - 200 OK, requête réussie ;
  - 201 OK, nouvelle ressource créée (commande POST) ;
  - 202 Requête acceptée, mais traitement incomplet ;
  - 204 OK, mais pas de contenu à renvoyer.
- 3XX : redirection, suite d'opérations à la charge du client :
  - 301 La ressource demandée a déménagé à une autre URL de façon permanente ;
  - 302 La ressource demandée a déménagé à une autre URL de façon temporaire ;
  - 304 Le document n'a pas changé (GET conditionnel)

Statut HTTP

- 4XX : erreur du client :
  - 400 Requête mal formulée ;
  - 401 Interdit, la requête nécessite une authentification ;
  - 403 Interdit, sans raison spécifique ;
  - 404 Non trouvé
- 5XX : erreur du serveur :
  - 500 Erreur interne du serveur ;
  - 501 Non implémenté ;
  - 502 Mauvaise passerelle ; réponse invalide d'une passerelle ;
  - 503 Service temporaire indisponible.

HTTP V1.1

- Meilleur gestion du cache ;
- En-tête **Host** devient obligatoire dans les requêtes ;
- Pour limiter les connections lors de l'envoi d'une page complexe, on envoie tout par la même connexion $\rightarrow$ pipeline $\Rightarrow$ accélère les communications et diminue la charge réseau.
- Supporte la négociation du contenu ;
- Nouvelles entêtes.

Bad Request

Requête du client HTTP

```
% telnet 127.0.0.1 80
GET / HTTP/1.1
```

Réponse du serveur HTTP

```
HTTP/1.1 400 Bad Request
Date: Tue, 02 Sep 2014 14:15:32 GMT
Server: Apache/2.4.6 (CentOS)
Content-Length: 226
Connection: close
Content-Type: text/html; charset=iso-8859-1

<!DOCTYPE HTML PUBLIC "-//IETF//DTD HTML 2.0//EN">
<html><head>
<title>400 Bad Request</title>
</head><body>
<h1>Bad Request</h1>
<p>Your browser sent a request that this server could not understand.<br
     />
</p>
</body></html>
Connection closed by foreign host.
```

Forbidden

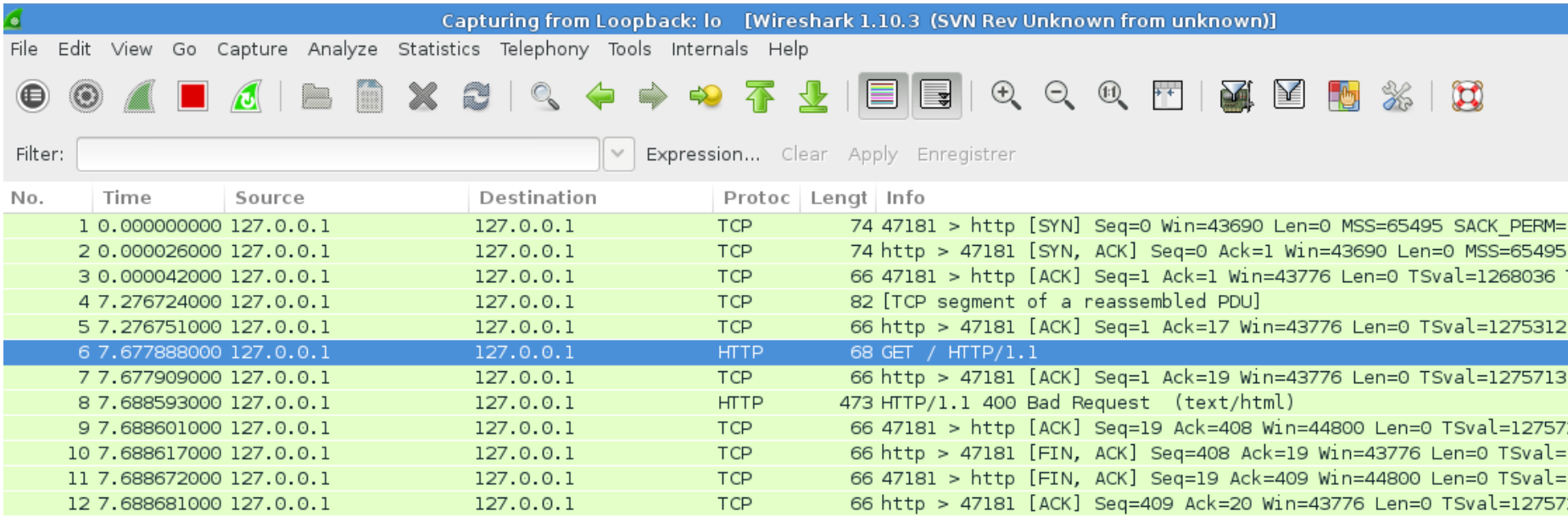
Requête du client HTTP

```
damien@ossus:4A/cours_reseau_2014_2015/cours_2% telnet 127.0.0.1 80
Trying 127.0.0.1...
Connected to 127.0.0.1.
Escape character is '^]'.
GET / HTTP/1.1
Host: localhost
```

Réponse du serveur HTTP

```
HTTP/1.1 403 Forbidden
Date: Tue, 02 Sep 2014 14:19:49 GMT
Server: Apache/2.4.6 (CentOS)
Last-Modified: Tue, 17 Jun 2014 16:00:47 GMT
ETag: "1310-4fc0a3f32a9c0"
Accept-Ranges: bytes
Content-Length: 4880
Content-Type: text/html; charset=UTF-8
```

Echange HTTP



Plan du cours

Préambule

Versions HTTP
- HTTP v0.9
- HTTP v1.0
- HTTP v1.1

Les fonctionnalités d'HTTP
- Les cookies
- HTTP et la sécurité
- Authentification
- Un peu d'admin..
- Autour d'HTTP

Suivi de Sessions avec HTTP (Session Tracking)

- Motivations :
  - La notion de session est importante dans une application conversationnelle de commerce électronique **j'ajoute ce produit à mon panier (existant)**

- Cependant HTTP est un protocole **stateless** : le serveur ne maintient pas d 'informations liées aux requêtes précédentes d'un même client. HTTP est donc **sessionless**
- Comment implanter la notion de session sur plusieurs requêtes HTTP documents, CGI, SSS, Servlet/JSP, ASP, PHP,

Suivi de Sessions avec HTTP (Session Tracking)

Méthodes

- Le serveur génère un identificateur de session et associe un état (et une date limite de validité) à une session ;
- Le client renvoie l'identificateur de session à chaque requête HTTP vers le serveur.

Échange et Stockage de l'identificateur de session

- Input HIDDEN dans les formulaires
- Réécriture des URLs (EXTRA_PATH)
- Cookies (désactivable)
- Identificateur de session SSL (Secure Socket Layer)

Suivi de session avec HTTP

Une session s'étend sur plusieurs requêtes documents

- CGI, SSS, Servlet, ASP
- le serveur maintient un contexte de session et y associe un identifiant de session

3 solutions de suivi

- input HIDDEN : contient l"identifiant de la session
- la Ré-écriture d'URL
- l'identifiant dans chaque URL (dans les documents)
- les Cookies
  - information positionnée par le serveur sur le client
  - la durée de vie du cookie dépasse la session
  - puis envoyé par le client à chaque requête

Positionnement des cookies

Requête :

```
GET /index.html HTTP/1.1
Host : www.exemple.org
```

Réponse :

```
HTTP/1.1 200 OK
Content-type : text/html
Set-Cookie : name=value
```

Suivant :

```
GET /spec.html HTTP/1.1
Host : www.exemple.org
Cookie : name=value
```

HTTPS

- Motivation :
  - Sécuriser (authentification, confidentialité) l'accès à un service Web SSL/TLS
- Phase 1 : Authentification du serveur et/ou du client par PKI
- Phase 2 : Chiffrage avec une clé (secrète) symétrique de session
- Phase 2bis : Reprise après déconnexion
- HTTP over SSL (TLS) :
  - URL : https ://host/document (Port TCP par défaut : 443)
  - HTTPS dans le JDK JSSE (javax.net.ssl.*) inclus dans J2SE 1.4
  - classe javax.net.HttpsURLConnection
  - Voir cours JCE
    (http ://www-adele.imag.fr/users/Didier.Donsez/cours/jce.pdf)
- Serveurs :Apache/SSL, iPlanet, MS IIS, OracleAS, IBM WebSphere
  ...

Introduction à la crypto

- Ce n'est pas un cours de crypto, mais vous devez comprendre !
- 2 types de crypto utilisés :
  - Chiffrement symétrique ;
  - Chiffrement asymétrique.

Introduction à la crypto

- Chiffrement symétrique :
  - Les deux correspondants utilisent la même clé pour chiffrer et déchiffrer les messages ;
  - Nécessite un canal de communication **sûr** si la clé est échangée sur le réseau ;
  - Généralement, les algorithmes sont rapides ;
- Enigma : créée par Turing ;
- Vigenère ;
- AES : Advanced Encryption Standard ;
- DES : Data Encryption Standard ;

Introduction à la crypto

- Chiffrement asymétrique :
  - Nécessite deux clés : une clé pour chiffrer et une clé pour déchiffrer ;
  - Je chiffre avec ma clé privée, vous déchiffrez avec ma clé public ;
  - Assure la non-répudiation des messages émis.
- RSA : Ronald Rivest, Adi Shamir et Leonard Adleman
- DSA : Digital Signature Algorithm

Secure Socket Layer

- un protocole à négociation : handshake SSL ;
- La connexion assure :
  - La confidentialité des données ;
  - Intégrité des données ;
  - Vérification de l'identité des correspondants ;
  - La connexion est fiable !
- SSL v1, v2, v3, TLS, etc.
- Authentification par certificat x509 ;
- Chiffrement des échanges par clés symétriques négociées.
- Faire un schéma !

Authentification dans HTTP

- Indiqué dans les ACL
- Mode d'authentification
  - BASIC (RFC 2617) :
    - nom d 'utilisateur et mot de passe échangé en clair (base64) !
    - base des mots de passe dans un fichier htpasswd utilitaires de gestion du fichier
  - DIGEST (RFC 2617)
    - sécurisation de BASIC
    - hachage sécurisé MD5 du (nom,password,URI, méthode,nombre aléatoire fourni par le serveur)

Authentification dans HTTP

- SSL/TLS
  - Secure Socket Layer (TLS : Transport Layer Security)
  - authentification avec CA du serveur (2.0) et du client (3.0)
  - confidentialité avec DES/AES, etc.
  - puis dialogue HTTP sur la connexion SSL
- Remarque : utilisation des cartes PKI pour l'identification du client

L'authentification applicative

- Motivations
  - interface de login
  - authentification plus forte (SSL)
  - Base d'identité
  - Fichier htpasswd, BD, Annuaire LDAP, ... (voir mod_auth_xxx de Apache)

L'authentification applicative

- L'application gère l'authentification de l'usager
  - formulaire d'accueil HTML (nom, password)
  - attention le mot de passe circule en "clair" : Utilisez une fonction JavaScript de hachage (MD5,SHA1) du mot de passe avec un nombre aléatoire pour éviter le "replay "
  - gestion des tables d'usager
  - une session est ensuite ouverte associé à un usager authentifié (ou non : par exemple rejet à bout de 3 tentatives)
- Remarque
  - TomCat peut de préciser les documents de login et d'erreur pour une arborescence de documents/servlets/JSP

Contrôle d 'Accès dans HTTP

- ACL (Access Control List) :
  - spécifie les autorisations (ALLOW) ou les interdictions (DENY) d'accès à une arborescence virtuelle du serveur
  - en fonction de l'authentification de la localisation du client, sous domaine DNS, réseau ou adresse IP
- ACF (Access Control File)
  - fichier regroupant les ACL
  - global : access.conf dans Apache
  - par arborescence : .htaccess
  - combinaison des ALLOW et des DENY

Audit des Requêtes

- Journaux des requêtes
  - les accès (access.log, refferee.log), et les erreurs (error.log), sont journalisés
- Exploitation des Journaux
  - erreur dans les liens, ...
  - clientèle, analyse d 'activité, ...
- Reporting (Présentation Synthétique)
- Pour Apache
  - AccessWatch, Wusage,Analog, wwwstat
- IIS, NS
  - intégré et visualisé par un script
- Généraux
  - Net Analysis (Net Genesis), Enterprise Suite (Web Trends)

Proxy

- Proxy
  - seul point de passage entre le réseau d'entreprise et l'extérieur (sécurité, firewall)
  - accès à des protocoles non implémentés par les clients Web ;
- Cache
  - soulager les accès externes (moins de bande passante)
- Miroir
  - réplication d'une base documentaire
  - améliorer le temps de réponse, soulager le réseau
- Robot
  - récupération online/offline d'une arborescence de documents
  - constitution d'un miroir local

Autour d'HTTP

- Mise à jour de sites
  - copie publique / copies de travail
- Portail
  - Point d'accès centralisant à un ensemble de sources Web
- Embedded Web Server
  - Serveur HTTP pour informatique embarquée

Proxy

- Fonctions
  - seul point de passage entre le réseau d'entreprise et l'extérieur
    - firewall, contrôler le profil d'utilisation, portail vers des ressources externes contrôlées (banques de données, ...)
  - accès à des protocoles non implémentés par les clients Web
  - WAIS, LDAP...
- Passerelle réseau de niveau applicatif

$\Rightarrow$ FIGURE

Fonctionnement Proxy

Fonctionnement du Proxy/Cache
figure

Cache Web

- But d'un cache Réseau
  - Usager
    - Améliorer les performances du browser en utilisant les documents présents dans le cache local
  - ISP (Internet Service Provider)
    - Soulager le réseau fédérateur en cachant les documents demandés par les usagers du sous-réseau
  - Provider Web
    - Soulager les serveurs applicatifs

Cache Web

- Documents
  - Textes (HTML, XML, ...) et Images fixes
  - Flux Audio et Video
  - têtes de réseau câble pour la Buffered-VOD
- Serveurs
  - Il fonctionne en mode Proxy
  - Squid, Harvest, Apache, MicroSoft, Sun, iPlanet, OracleAS WebCache, WebSphere