Aide à la Détection de Faiblesses d'un site Web
Mandataire inverse, Modsecurity

S. Aicardi

Journées Mathrice, Angers, 17-19 Mars 2009

Serveur mandataire (Proxy)

C'est un serveur utilisé comme intermédiaire entre des clients et des serveurs. Au lieu d'adresser sa requète directement au serveur final, le client la transmet au mandataire qui effectue la requète au serveur final, filtre ou reformatte éventuellement la réponse avant de la transmettre au client.

Serveur mandataire (Proxy)

Buts :

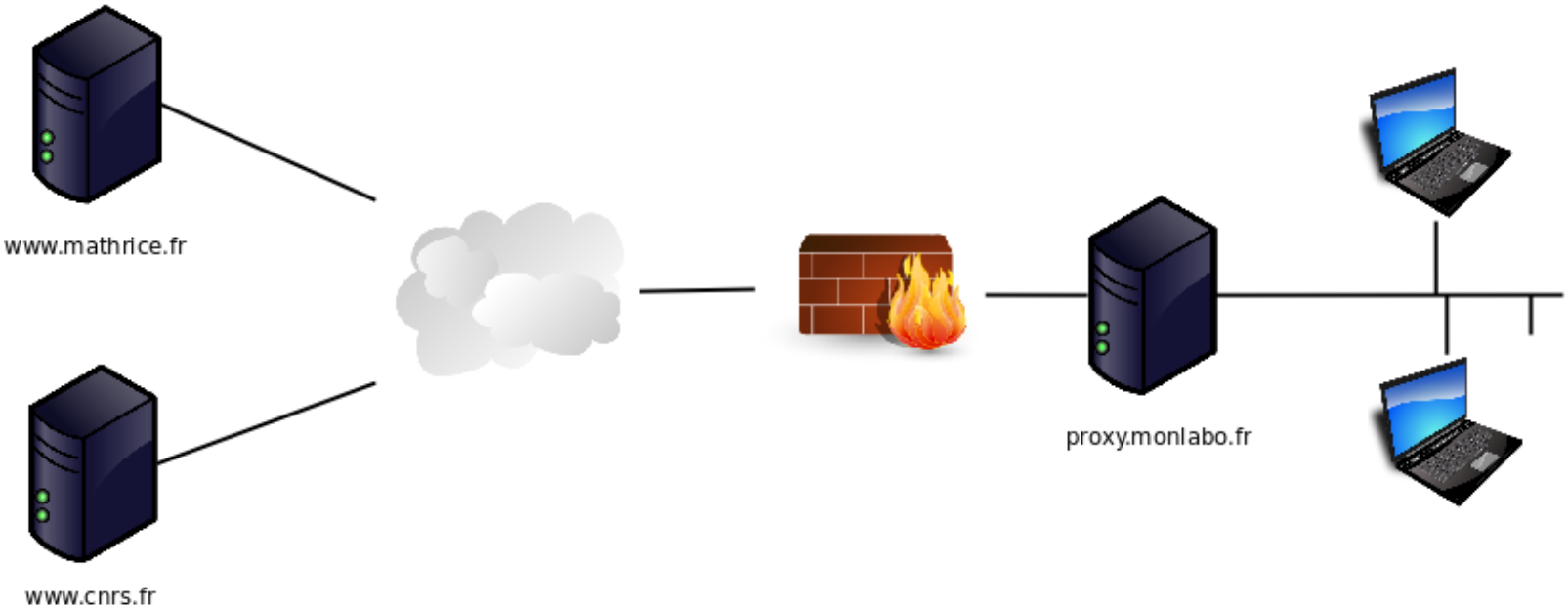
- Cache : le mandataire peut stocker les réponses pour le retransmettre directement à la prochaine requète identique.
- Filtrage applicatif : le mandataire peut supprimer un contenu non autorisé ou hostile dans sa réponse au client
- Anonymat/sécurisation du réseau
- Journalisation des requètes (LCEN)

Serveur mandataire (Proxy)

Implémentations :

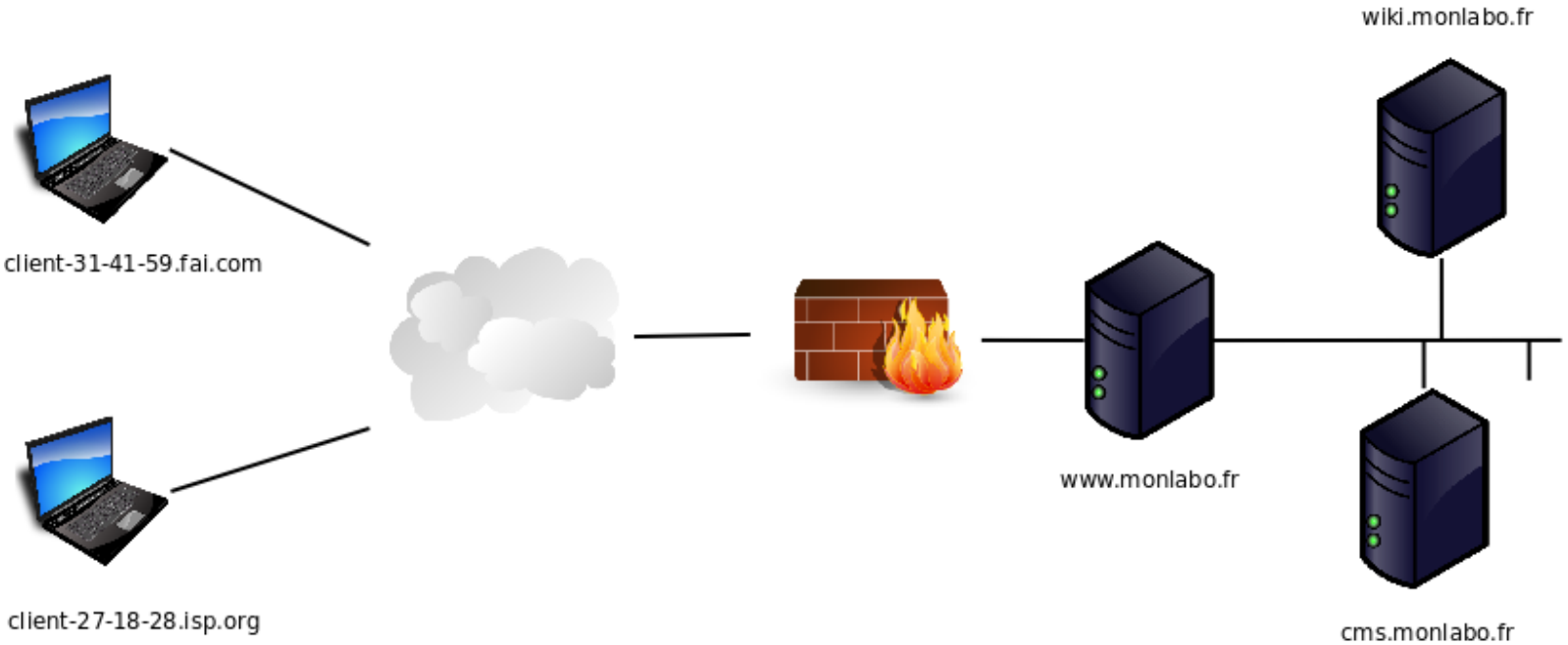
- Squid est le serveur mandataire libre le plus utilisé pour les protocoles HTTP, HTTPS et FTP.
- Apache peut servir de mandataire
- SOCKS5 est un protocole de mandataire générique multiprotocole (TCP, UDP). Dante Socks Server est un serveur libre implémentant SOCKS5. OpenSSH ou putty permettent également de créer un serveur SOCKS5.

Serveur mandataire et réseau



Mandataire inverse

Il s'agit d'un mandataire monté « à l'envers », le client étant quelque part dans l'Internet et le serveur étant dans l'Intranet.



Mandataire inverse

Buts:

- Point d'entrée unique (un seul port 80 à ouvrir vers Internet) $\Longrightarrow$ journalisation plus efficace ;
- Accélération de requètes aux sites dynamiques (Zope...) ;
- Répartition de charge entre plusieurs serveurs web ;
- Cloisonnement des services web (wiki, cms, agenda, etc.) ;
- Filtrage et réécritures des requètes ($\Longrightarrow$ protection en amont contre les failles) ;
- Occultation de la topologie interne et "enfouissement" des serveurs web et des bases de données.

Mandataire inverse

Inconvénients :

- Point d'entrée unique $\Longrightarrow$ s'il tombe, tous les sites tombent ;
- Complexification du réseau ;
- Rupture des communications HTTPS qui ne peuvent plus se faire de bout en bout (par ailleurs pb https et virtual hosts) :
- Gestion des redirections fastidieuse et source d'erreurs.

Mandataire inverse

Implémentations :

- Squid peut être utilisé en mandataire inverse ;
- Varnish est un nouveau projet de mandataire inverse ;
- Pound est spécialisé dans la répartition de charge ;
- Apache peut être configuré en mandataire inverse.

Utilisation d'apache en mandataire inverse

Apache est sans doute déjà en place comme serveur web $\Longrightarrow$ cela facilite la transition.
Voici le minimum vital pour que `www.monlabo.fr` présente les sites `wiki.monlabo.interne` et `cms.monlabo.interne` :

```
LoadModule proxy_module modules/mod_proxy.so
LoadModule proxy_http_module modules/mod_proxy_http.so

ProxyRequests Off

<Proxy *>
Order deny,allow
Allow from all
</Proxy>

ProxyPass /wiki http://wiki.monlabo.interne
ProxyPassReverse /wiki http://wiki.monlabo.interne
ProxyPass /cms http://cms.monlabo.interne
ProxyPassReverse /cms http://cms.monlabo.interne
```

Utilisation d'apache en mandataire inverse

Pour les configurations plus complexes, on peut utiliser `mod_rewrite` :

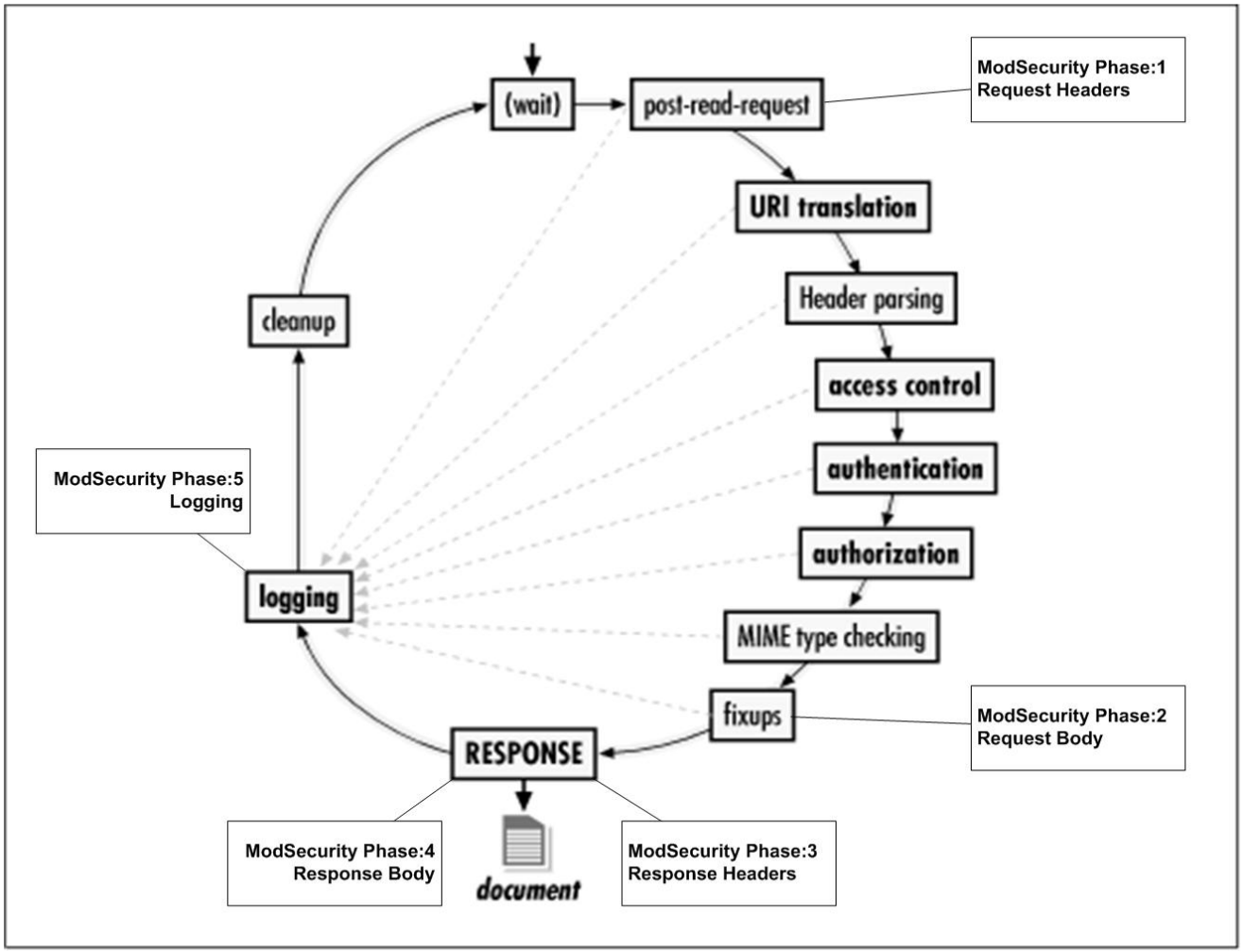
```
LoadModule proxy_module modules/mod_proxy.so
LoadModule proxy_http_module modules/mod_proxy_http.so
LoadModule rewrite_module modules/mod_rewrite.so

RewriteEngine On
RewriteRule ^/wiki/(.*)$ http://wiki.monlabo.interne/$1 [P]
ProxyPassReverse /wiki http://wiki.monlabo.interne
RewriteRule ^/cms/(.*)$ http://cms.monlabo.interne/$1 [P]
ProxyPassReverse /cms http://cms.monlabo.interne
```

Modsecurity

`Modsecurity` est un module de pare-feu applicatif pour Apache.
Il protège les sites webs d'une collection d'attaques classiques (injections PHP ou SQL, exposition de fichiers sensibles, etc.)

Modsecurity : principe de fonctionnement



Modsecurity : configuration d'Apache

Après installation de `Modsecurity`, il faut insérer les lignes suivantes dans la configuration d'Apache :

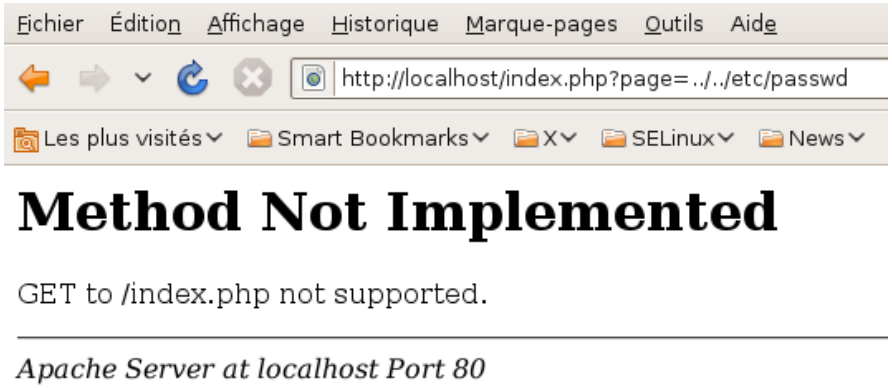
```
LoadFile /usr/lib/libxml2.so.2
LoadModule security2_module modules/mod_security2.so
Include PATH_vers_les_regles/*.conf
```

Les fichiers de règles contiennent des listes du type suivant :

```
  SecRule REQUEST_FILENAME|ARGS|ARGS_NAMES ‘‘(?:\b(?:\.(?:ht(?:access|passwd|group)|www_?acl)
|global\.asa|httpd\.conf|boot\.ini)\b|\/etc\/)’’ \
  ‘‘phase:2,capture,t:none,t:htmlEntityDecode,t:lowercase,ctl:auditLogParts=+E,
deny,log,auditlog,status:501,msg:’Remote File Access Attempt’,id:’950005’,
tag:’WEB_ATTACK/FILE_INJECTION’,logdata:’%{TX.0}’,severity:’2’’’
```

Modsecurity : test

Côté client :



Côté serveur :

```
Message: Access denied with code 501 (phase 2). Pattern match ‘‘(?:\b(?:\.(?:ht(?:access|passwd|
group)|www_?acl)|global\.asa|httpd\.conf|boot\.ini)\b|\/etc\/)’’ at ARGS:page. [file ‘‘/etc/apac
he2/mod-security-rules/modsecurity_crs_40_generic_attacks.conf’’] [line ‘‘114’’] [id ‘‘950005’’]
 [msg ‘‘Remote File Access Attempt’’] [data ‘‘/etc/’’] [severity ‘‘CRITICAL’’] [tag ‘‘WEB_ATTACK
/FILE_INJECTION’’]
Action: Intercepted (phase 2)
```

Modsecurity et mandataire inverse

Modsecurity peut d'installer sur un serveur Apache hébergeant du contenu dynamique, mais le module prend tout son sens sur un mandataire inverse. On protège alors tous les services webs d'un seul coup.

Google/Yahoo! Hacking

Quelques liens amusants :

- des serveurs qui publient leurs logs !

  http://www.google.com/search?q=inurl%3Alogs%2Faccess.log

  http://search.yahoo.com/search?p=inurl%3Alogs%2Faccess.log
- des serveurs SPIP qui publient leurs données internes

  http://www.google.com/search?q=inurl%3Ameta_cache.txt
- des listes de mots de passe

  http://www.google.com/search?q=intitle%3A"Index+of"+".htpasswd"+htpasswd.bak
- des configurations PHP

  http://www.google.com/search?q=intitle%3Aphpinfo+"PHP+Version"
- des configurations de proxy

  http://www.google.com/search?q=inurl%3Aproxy+ext%3Apac+findproxyforurl