M2 RADIS : Approfondissement Réseaux

`apache2` en Reverse Proxy et `mod_security`

Cette séance a un double objectif :
. Acquérir les compétences suffisantes afin d'installer et configurer un reverse proxy apache2
. Sécuriser avec un minimum d'effort des serveurs web internes (apache2) directement ou via un reverse proxy : implementation du module mod security 2.5 (ou 2.6 en debian backports)

1 Préparation du TP

1.1 Description

La figure 2 représente un réseau que vous pourriez utiliser. Vous avez à votre disposition 1 réseau entre `11` et `1F` en hexadécimal. Il sera noté `XX` dans le reste de l'énoncé. Le numéro `X` de votre réseau IPv4 en est la transformation en décimal. Par exemple si `XX = 0A`, votre numéro de réseau IPv4 serait `10`. Votre zone DNS forward serait `zone0A`.

En cas de réutilisation d'un fichier `.mar` de base, l'adresse MAC de **m1** doit *impérativement* être changée avec `XX` comme dernier mot afin d'éviter des collisions au niveau du bridge avec d'autres instances de marionnet.

Si la quantité de configuration vous fait peur, vous pouvez utiliser le schéma simplifié.

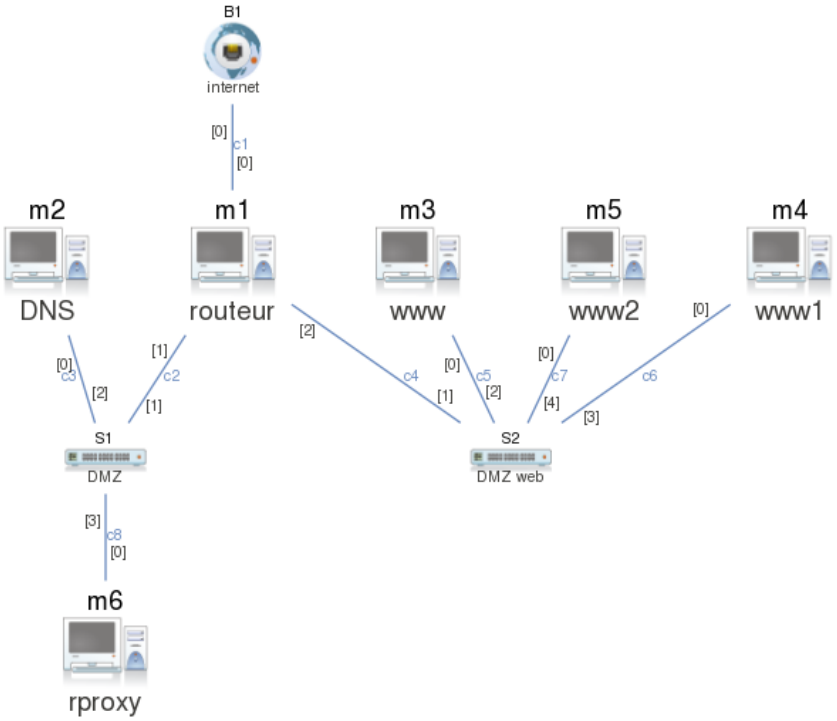

FIGURE 1 – Schéma du réseau

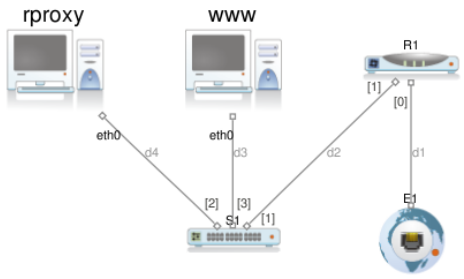

FIGURE 2 – Schéma simplifié

1.2 Configuration réseau

L'adresse IP de l'interface externe (eth0) de m1 doit être `192.168.128.X` pour que le réseau sous-jacent soit joignable par les autres instances de marionnet.

La seule adresse IP imposée est celle du service DNS (`192.168.X.2`) car elle est déclarée au niveau de la délégation de zone.

L'adressage de la zone `DMZ web` est laissé à votre discrétion. Il faudra configurer les autres machines en conséquence.

Aucun traffic direct ne devrait être autorisé entre le bridge et la zone `DMZ web`. Le traffic web vers les services de la zone `DMZ web` doit être assuré par la machine `rproxy`. Comment mettre en place cette configuration ?

1.3 Configuration DNS

Comment configurer la résolution de nom des services web. Par exmple, que doit indiquer `host -t ANY www.zoneX.tp.info.unicaen.fr` ?
Vous pouvez aussi remplir « à la main » le fichier /etc/hosts de chaque machine mais c'est fastidieux et source d'erreurs.

1.4 Configuration des services web

Vérifiez la présence des services apache des machines `m3, m4, m5`. Vous pouvez configurer plusieurs serveurs sur la même machine basés sur des IP différentes ou sur des noms différents mais c'est hors du sujet de ce TP. Faites vivre un peu ces services web. Vous pourrez par exemple faire une page d'accueil avec un formulaire, mettre quelques documents à télécharger ...

2 Apache en reverse proxy

2.1 Informations utiles

Après la modification d'un paramètre d'apache, pensez à faire un `reload` ou `restart` en fonction des cas. Pensez à regarder les logs de `/var/log/apache2/` ou tout autre fichier que vous aurez défini dans votre configuration d'apache.

Faites attention aux paramètres de vos navigateurs : un serveur proxy ou le cache interne du navigateur peuvent vous jouer des tours. Utlisez des navigateurs *légers* : links, wget, dillo ou dont vous maîtrisez parfaitement les réglages. En cas de doute, parlez le `HTTP/1.0`, par exemple :

```
telnet 192.168.XX.33 80
GET http://www.zoneX.tp.info.unicaen.fr/ HTTP/1.0
```

2.2 Ajout des modules nécessaires

La configuration standard `debian` n'est pas adaptée à un usage en reverse proxy. Quelques changements sont à effectuer autour des modules chargés et de leur configuration.

2.3 La configuration du proxy

Le site par défaut contiendra une simple page html disant que le site n'est pas accessible quelle que soit la requête effectuée.
Une fois le site par défaut installé, mettez en place l'accès aux sites suivants en utilisant les serveurs internes exécutés par les machines `www`). Par exemple :
- `http://www.zoneX...` via le serveur apache2 de www.
Testez l'accès effectif à ces sites. Éventuellement depuis un autre réseau que le votre.
Mettez ensuite en place un alias et expérimentez le contrôle d'accès par le contrôle des règles de réécriture (`RewriteCond`).
- Redirection de `http://www.zoneX.tp.info.unicaen.fr` vers un autre serveur
- Récupération des fichiers images depuis un autre serveur (www2 ?)
- Exécution des scripts php depuis un autre serveur (www3 ?)
- Restreindre l'accès d'un des serveurs en fonction de l'IP d'accès

à faire après `mod_security`

Sur l'un des service www, vous pourrez activer un accès direct à un script analysant un formulaire via une url simplifiée de la forme `http://www/username/chaine` au lieu de `http://www/index.php?username=chaine`. D'autre part, la tentative d'accéder à `username=admin` pourra rediriger vers un autre utilisateur.
Pour ceux qui ont le temps et qui veulent s'amuser un peu, il y a des tas de choses à lire et à essayer : `http://httpd.apache.org/docs/2.2/mod/mod_rewrite.html`

3 Configuration de `mod_security 2.5`

3.1 Introduction

La distribution debian squeeze utilise actuellement la version 2.5 de mod_security. Attention, la configuration de la version 2.6 étant légèrement différente, ce qui est expliqué ici ne fonctionnera pas sans quelques ajustements sur une debian wheezy ou avec le paquet issu des backports pour squeeze.
Les paquets suivants ont été préalablement installés :

```
apt-get install libapache-mod-security
```

L'installation par défaut ne charge quasiment aucune règle.
Il est nécessaire d'initialiser quelques répertoires et fichiers avant de continuer. Copions certaines règles par défaut de `mod_security2` (ou toutes avec '*') :

```
mkdir /etc/apache2/rules.d
cp -a /usr/share/doc/mod-security-common/examples/rules/base_rules/* \
  /etc/apache2/rules.d
```

Il faut au moins indiquer l'emplacement des fichiers de log :

```
# rules.d/modsecurity_crs_10_config.conf
SecDebugLog /var/log/apache2/modsec_debug.log
SecAuditLog /var/log/apache2/modsec_audit.log
```

Le fichier de chargement du module `mod_security2` pour apache devrait avoir le contenu approchant :

```
# /etc/apache2/mods-available/mod_security2.load
LoadFile /usr/lib/libxml2.so
LoadFile /usr/lib/liblua5.1.so.0
LoadModule security2_module /usr/lib/apache2/modules/mod_security2.so
```

Et l'un des fichiers de configuration d'apache2 devrait référencer le chargement du répertoire de règles de `mod_security2.conf` :

```
Include /etc/apache2/rules.d/*.conf
```

Maintenant nous allons charger le module dans la configuration apache afin que celui-ci utilise le module `mod_security2`.

```
a2enmod mod_security2
```

Comme d'habitude, il faut recharger apache. Étant donné la dépendance sur des fichiers `.so` externe, c'est plus probablement un restart qu'il faut effectuer.
Vous pouvez vérifier que le module fonctionne bien :

```
wget "http://localhost/index.php?username=' or 1 = 1"
wget http://www.zoneX/telnet.exe
# ou directement :
telnet www.zoneX 80 puis GET /telnet.exe
...
```

Vérifiez ensuite la règle appliquée dans les logs.
Le module `mod_security` offre par défaut une protection contre les attaques les plus courantes. Toutefois il est important de ne pas faire une confiance aveugle aux règles par défaut sans y avoir prêté attention.

Vous pouvez remarquer que `mod_security` ne filtre pas. L'injection SQL aurait parfaitement fonctionné s'il y avait une vraie application derrière (ici nous avons un 404).
En examinant les logs, vous constatez que `mod_security` signale la possible injection SQL mais la laisse passer (`PASS`). *Moralité : il faut rester vigilant sur l'application des règles par défaut de* `mod_security` *et continuer à bien surveiller les logs.*
Afin d'activer le filtrage sur toutes les requêtes identifiées par les règles (et pas seulement les logguer) décommenter la ligne 107 du fichier `modsecurity_crs_10_config.conf` (" SecDefaultAction ").

```
SecDefaultAction "phase:2,log,deny,status:403"
```

Vous pouvez faire de nouveau le test d'injection sql, vous constaterez que vous êtes dorénavant filtré par `mod_security`.
Le cas échéant, il faudra configurer `php5` et écrire un petit script à exploiter. Par exemple :

```
apt-get install libapache2-mod-php5

<?php
  echo strip_tags($_GET['username'];
?>
```

4 Fichiers de configuration

4.1 apache

Les modules à charger

```
a2enmod rewrite
a2enmod proxy
a2enmod proxy_http
```

Le fichier de configuration interdisant l'accès au proxy par défaut pourra être modifié pour y voir (par exemple) :

```
# /etc/apache2/mods-available/proxy.conf
ProxyRequests On
Allow from all
```

4.2 sites

Interdiction par défaut.

```
NameVirtualHost 192.168.128.X
DocumentRoot /var/www/rproxy
<VirtualHost 192.168.128.X>
  RewriteEngine On
  RewriteRule   ^.* /erreur.jpg [L]
</VirtualHost>
```

Site public.

```
<VirtualHost 192.168.128.X>
  ServerName www.zoneX.tp.info.unicaen.fr
  ServerAlias www

  RewriteRule ^/(.*)$ http://www-intra.zoneX.tp.info.unicaen.fr:50000/$1 [P]
</VirtualHost>
```

Site des photos.

```
<VirtualHost 192.168.128.X>
  ServerName www-photos.zoneX.tp.info.unicaen.fr
  ServerAlias www-photos
  RewriteEngine On
  RewriteRule ^/(.*)$ http://www-intra.zoneX.tp.info.unicaen.fr:50001/$1 [P]
</VirtualHost>
```

Restrictions d'accès. Attention, les `IP` ne correspondent peut-être pas.

```
<VirtualHost 192.168.128.X>
  ServerName www-restreint.zoneX.tp.info.unicaen.fr
  ServerAlias www-restreint

  RewriteEngine On
  RewriteCond %{REMOTE_ADDR} ^192\.168\.X\. [OR]
```

```
  RewriteCond %{REMOTE_ADDR} ^127\.0\.0\.1
  RewriteRule ^/(.*)$ http://www-intra.zoneX.tp.info.unicaen.fr:50002/$1 [P]

  RewriteRule     ^.* /interdit.jpg [L]
</VirtualHost>
```

Redirection des photos.

```
<VirtualHost 192.168.128.X>
  ServerName www-redirect.zoneX.tp.info.unicaen.fr
  ServerAlias www-redirect
  RewriteEngine On
  RewriteRule ^/photos(.*) http://www-photos.zoneX.tp.info.unicaen.fr [R]
  RewriteRule ^/(.*)$ http://www-intra.zoneX.tp.info.unicaen.fr:50000/$1 [P]
</VirtualHost>
```

Pour mémoire les `RewriteFlags` :

```
  R : rewrite
  P : proxy
  L : last rule
```

Les 2 questions à faire après `mod_security`. Que se passe-t-il en fonction de l'ordre d'écriture de ces règles ?

```
    RewriteEngine On
    RewriteRule ^/u/(.*)$ /index.php?user=$1
    RewriteCond %{QUERY_STRING} user=admin
    RewriteRule ^/(.*)$ http://www/$1?username=realadmin [P]
```

Code html+php permettant de faire une réponse générique lors de l'échec de l'affichage d'un site :

```
<?php
  $srv = $_SERVER['HTTP_HOST'];
?>
<?xml version="1.0" encoding="utf-8" ?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN" "DTD/xhtml1-strict.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="fr" lang="fr">
<head>
<meta http-equiv="Content-Type" content="text/html;charset=utf-8" />
<title><?php echo $srv; ?></title>
</head>
<body>
<h1>Ooops ...</h1>
<p>
The requested site ( <?php echo $srv; ?> ) is nonexistant or temporarily unavailable.<br />
</p>
</body>
</html>
```

Malgré le soin apporté à ce document, il se peut que des coquilles subsistent. D'autre part, les exemples proposés dans ce document ne dispensent pas de la lecture des pages de manuel adéquates.