Restrictions d'accès

> Because now, Detective Inspector Bishop, there is no power on this earth that can stop me !
>
> Le dixième docteur

Il est parfois nécessaire de restreindre l'accès à une ressource afin de s'assurer que seules les personnes autorisées consultent la ressource.

Une méthode simple consiste à prendre en compte l'adresse IP du client afin de lui autoriser ou refuser l'accès à la ressource.

1 Apache

Certaines directives et conteneurs permettent de définir des restrictions d'accès à tous les fichiers d'un répertoire, *via* les conteneurs `<Directory>` ou *via* le fichier `.htaccess` (ce type de restriction ne peut pas être associé directement à un fichier donné). Ces directives sont fournies par des modules `mod_auth*_*`, dont un sous-ensemble est généralement activé par défaut par les distributions.

Dans Apache 2.2 (que vous rencontrerez peut-être un jour ou l'autre), les restrictions sont mises en place au moyen des directives `Order`, `Allow` et `Deny`. Ces directives sont obsolètes avec Apache 2.4, elles sont donc à éviter (bien qu'on puisse les utiliser grâce au module `mod_access_compat`).

La directive permettant les restrictions est, depuis Apache 2.4, la directive `Require`. Les modules `mod_authz_core` et `mod_authz_host` mettent à disposition des fournisseurs d'autorisation génériques utilisables avec la directive Require.

On les utilise ainsi :

```
Require *option* *fournisseur d'autorisation* *arguments*
```

Parmi les fournisseurs d'autorisation, citons :

- `all`, qui prend en argument `granted` ou `denied` pour autoriser ou bloquer toutes les requêtes ;
- `ip`, qui prend une ou plusieurs adresse IP ou réseaux ;
- `host`, qui prend en argument tout ou partie d'un nom d'hôte qui sera comparé au nom d'hôte du client via une double interrogation DNS (à éviter) ;
- `local`, qui ne prend pas d'argument, et qui autorisera l'accès si le client est la machine locale (`127.0.0.0/8`, `::1` ou si l'IP du client et du serveur sont les mêmes) ;
- `method`, qui prend en argument une ou plusieurs méthodes HTTP.

Il est à noter que pour les fournisseurs d'autorisation `ip` et `host` peuvent prendre en argument des adresses IP partielles (équivalentes à des adresses réseaux) ou des noms d'hôte partiels (c-à-d que le nom d'hôte du client finit comme le paramètre de la directive) :

```
Require ip 192.168
Require host .net
```

L'adresse partielle `192.168` correspond au réseau `192.168.0.0/16`, qu'il est possible de noter `192.168.0.0/255.255.0.0` (`ip` accepte ces trois syntaxes).

Pour inverser une requête, on utilisera l'option `not` [1] :

1. sauf pour `all`, puisqu'on a l'argument `denied`

```
Require not local
```

Notez que `not` étant la négation d'une valeur, il ne peut pas être utilisé pour autoriser ou interdire une requête, car *non vrai* ne sera pas interpreté par Apache comme *faux*. Ainsi, pour interdire la visite d'une page à l'aide d'une négation, le bloc doit contenir un élément évalué à vrai ou faux.

Pour grouper différentes directives d'autorisation, on pourra utiliser les conteneurs `<RequireAll>` (toutes les directives doivent correspondre pour autoriser l'accès), `<RequireAny>` (au moins une directive doit correspondre) ou `<RequireNone>` (aucune directive ne doit correspondre).

Si les directives ne sont pas dans un conteneur `<Require*>`, on considérera qu'elles sont dans un conteneur `<RequireAny>`.

Vous trouverez un peu de documentation sur https://httpd.apache.org/docs/2.4/fr/howto/access.html.

Exemples :

```
# La ressource n'est accessible qu'au réseau 192.168.1.0/24
<Directory /var/www/lan>
    Require ip 192.168.1.0/24
</Directory>
# La ressource est interdite au réseau 10.10.0.0/16
<Directory /var/www/other_lan>
    <RequireAll>
        Require all granted
        Require not ip 10.10.0.0/16
    </RequireAll>
</Directory>
```

2 Nginx

C'est le module `ngx_http_access_module` [2] qui fournit les directives `allow` et `deny`, auquelles on donnera un paramètre (une adresse IP, un réseau au format CIDR ou `all`).

```
deny 192.0.2.1;
allow 192.0.2.0/24;
allow 198.51.100.0/24;
deny all;
```

Les directives sont évaluées de haut en bas : la première qui correspond s'applique. Attention donc à ne jamais placer un `allow all;` ou un `deny all;` en haut de la liste, les autres directives d'accès ne seraient jamais évaluées !

2. http://nginx.org/en/docs/http/ngx_http_access_module.html