Contrôles d'accès

I'm not a hero.

Le douzième docteur

Nous avons déjà étudié précédemment quelles étaient les possibilités de restrictions en fonction de l'origine d'un client (directives `Require` pour Apache et `allow` et `deny` pour Nginx). D'autres directives permettent de contrôler l'accès à une page ou à un ensemble de pages.

Le protocole d'authentification HTTP prévoit 2 méthodes d'authentification différentes : le mode *basic* (où les informations correspondantes transitent en clair) et le mode *digest* (où les informations correspondantes sont chiffrées).

NB : Nginx ne supporte pas le mode *digest*, uniquement le *basic*.

1 Différences entre authentification *basic* et *digest*

Dans l'authentification basique, le couple identifiant / mot de passe est envoyé encodé en base64 (et donc décodable) dans l'en-tête `Authorization`. Ainsi, pour l'utilisatrice `dr` avec le mot de passe `who`, on enverra :

```
Authorization: Basic ZHI6d2hv
```

(`ZHI6d2hv` correspond à `dr:who` encodé en base64 : `echo -n dr:who | base64`)

Les informations envoyées dans l'authentification *digest* dépendent de paramètres envoyés par le serveur et ne contiennent pas de données décodables. Exemple :

```
Authorization
    Digest username="luc", realm="foo",
        nonce="ZklQHRbRBQA=caebe745c92bc7932ad1b071631990d13cda189d",
        uri="/digest/",
        algorithm=MD5,
        response="98cee1c3b1eb9733695538b0d6e464d9",
        qop=auth,
        nc=00000006,
        cnonce="848b920ed0422c15"
```

Si l'authentification *digest* semble plus robuste, elle ne suffit pas pour autant améliorer la sécurité de manière significative par rapport à l'authentification basique. En outre, le stockage du mot de passe sur le serveur est encore moins sûr dans le cas d'une authentification à base de condensé que dans le cas d'une authentification basique. C'est pourquoi l'utilisation de l'authentification basique associée à un chiffrement de la connexion via SSL/TLS constitue une bien meilleure alternative.

2 L'authentification basique

Cette méthode ne chiffrant pas les informations d'authentification, celle-ci ne doit être mise en place en production que sur un serveur sécurisé via le protocole HTTPS (sauf pour des tests, bien sûr).

Pour ce premier mode, il faut tout d'abord créer un fichier texte contenant les logins et les mots de passe. Le logiciel `htpasswd` (disponible sur Debian via le paquet `apache2-utils`) peut être utilisé pour cela.

Il accepte en particulier les options suivantes :

— `-c` : création d'un nouveau fichier ;
— `-b` : considère le mot de passe de la ligne de commande au lieu de le demander interactivement (utile lorsque l'on veut automatiser la création des comptes).

Pour créer le fichier :

```
htpasswd -c fichier nom_de_l_utilisatrice
```

Pour modifier le fichier :

```
htpasswd fichier nom_de_l_utilisatrice
```

Pour ajouter une utilisatrice au fichier :

```
htpasswd fichier nom_de_l_autre_utilisatrice
```

Une fois le fichier des mots de passe créé, il est alors nécessaire de configurer le serveur Web pour l'utiliser.

2.1 Apache

L'utilisation du fichier de mots de passe se fait dans les contextes *répertoire* ou *.htaccess* à l'aide de plusieurs directives.

```
<Location /private>
    AuthName "closed site"
    AuthType basic
    AuthBasicProvider file
    AuthUserFile /usr/local/apache2/auth/userfile
    Require user user1 user2 ...
</Location>
```

Dans l'exemple ci-dessus, l'authentification en elle-même est assurée par les 4 premières lignes, commençant toutes par `Authxxx`. La dernière ligne (`Require`) est utilisée une fois l'authentification effectuée. C'est une directive d'*autorisation*, détaillant les utilisatrices autorisés [1]. Nous aurions pu choisir d'autoriser toutes les utilisatrices du fichier plutôt que quelques-unes avec `Require valid-user`.

Les directives d'authentification (`Authxxx`) sont fournies par différents modules Apache (via des fichiers, une base de données, un annuaire LDAP…). Regardez les modules dont le nom commence par `mod_auth` sur https://httpd.apache.org/docs/2.4/mod/.

Notez bien que l'authentification basique ne se rapporte pas à la méthode d'authentification sous-jacente (fichier, base de données, annuaire LDAP…) mais à la manière dont se fait la recherche

— `AuthName` sert généralement à donner une indication à la personne voulant s'authentifier… ou pas. Mettez ce que vous voulez.
— `AuthType` sert à indiquer le module d'authentification utilisé : `basic` dans notre cas.
— `AuthBasicProvider` est une directive propre au module `mod_auth_basic` (que nous avons choisi via `AuthType basic`) et indique la source de données à utiliser pour l'authentification : fichier, base de données, annuaire LDAP, etc.
— `AuthUserFile` est une directive propre au module `mod_authn_file`, que nous avons désigné comme fournisseur de la source de données pour l'authentification avec `AuthBasicProvider file`. Il s'agit du chemin du fichier contenant les identifiants des utilisatrices.

1. http://httpd.apache.org/docs/2.4/mod/core.html#require

2.2 Nginx

Le support de l'authentification est plus sommaire dans Nginx. Ainsi, il n'est pas possible de spécifier un sous-ensemble des utilisatrices autorisées :

```
location / {
    auth_basic           "closed site";
    auth_basic_user_file conf/htpasswd;
}
```

- `auth_basic` peut prendre une chaîne de caractères en argument ou `off` (pour désactiver l'authentification).
- `auth_basic_user_file` prendra en argument le chemin du fichier des utilisatrices et de leurs mots de passe. Cet argument peut inclure des variables (comme par exemple `/etc/nginx/auth/$host`).

Le module fournissant l'authentification est `ngx_http_auth_basic_module`[2].

Notez que contrairement à Apache, on ne peut pas autoriser que quelques utilisatrices : toutes les utilisatrices présentes dans le fichier sont autorisées.

3 L'authentification *digest*

Si le principe de fonctionnement de l'authentification *digest* est différent de la méthode *basic*, son principe de mise en œuvre est relativement similaire. Vous trouverez plus d'informations sur https://httpd.apache.org/docs/current/mod/mod_auth_digest.html.

L'utilisation de ce mode d'authentification étant désuet depuis la généralisation du web sécurisé, nous ne nous étendrons pas plus sur l'authentification *digest*.

4 Pour aller plus loin

La page https://httpd.apache.org/docs/2.4/howto/auth.html présente une synthèse des solutions d'authentification HTTP avec Apache (l'utilisation d'un fichier n'a rien d'obligatoire, on peut utiliser une base de données ou un serveur LDAP par exemple), d'autorisations et de contrôle d'accès.

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