Configurer l'accès Subversion par HTTP avec Apache

Stéphane Bortzmeyer
<stephane+blog@bortzmeyer.org>

Première rédaction de cet article le 22 novembre 2008

https://www.bortzmeyer.org/access-subversion-apache.html

Le VCS Subversion est probablement désormais le plus utilisé des VCS centralisés, une position qui avait été longtemps tenue par CVS. Une de ses particularités est la possibilité d'accéder au dépôt Subversion par différents protocoles réseaux. Pour HTTP, le moyen le plus courant est d'utiliser Apache.

Le client Subversion désigne en effet un dépôt par un URL. Cet URL peut désigner un fichier local (il commence alors par `file://`, cf. RFC 8089[1]), un accès via un compte distant atteint en SSH et il commence alors par `svn+ssh://` ou bien un dépot distant accédé en HTTP, méthode la plus courante dès que le dépôt est utilisé par plusieurs utilisateurs.

Subversion utilise pour cela le protocole WebDAV, une extension à HTTP, normalisé dans le RFC 4918, plus des extensions spécifiques à Subversion. On va utiliser Apache, comme documenté en <http://svnbook.red-bean.com/en/1.5/svn.serverconfig.httpd.html>. Mettons qu'on veuille un dépôt accessible par l'URL `https://svn.example.net/`, qui permette les accès anonymes mais qui réserve certains répertoires à des utilisateurs bien identifiées (via LDAP). Apache ne demandera une authentification que pour les parties du dépôt où c'est nécessaire. La configuration va ressembler à :

```
# Module Apache pour WebDAV
LoadModule dav_module /usr/lib/apache2/modules/mod_dav.so
# Module Apache pour les extensions Subversion à WebDAV
LoadModule dav_svn_module /usr/lib/apache2/modules/mod_dav_svn.so
# Module Apache pour l'authentification par répertoire dans le dépôt
# (ce qu'Apache ne sait pas faire tout seul)
LoadModule authz_svn_module /usr/lib/apache2/modules/mod_authz_svn.so

# Uniquement en httpS, pour des raisons de sécurité, donc port 443
<VirtualHost *:443>
    ServerAdmin webmaster@example.net
    ServerName svn.example.net
```

1. Pour voir le RFC de numéro NNN, `https://www.ietf.org/rfc/rfcNNN.txt`, par exemple `https://www.ietf.org/rfc/rfc8089.txt`

```
    # Pour le chiffrement, on utilise GNU TLS (et pas OpenSSL)
    GnuTLSEnable  on
    GnuTLSCertificateFile /etc/ssl/certs/ssl-cert-www.example.net.pem
    GnuTLSKeyFile /etc/ssl/private/ssl-cert-www.example.net.key
    GnuTLSPriorities NORMAL:!AES-256-CBC:!DHE-RSA

 <Location />

  # Le dépôt
  DAV svn
  SVNPath /home/Subversion-Repository

  # Couper l'authentification ordinaire et activer celle par
  # LDAP. Naturellement, d'autres méthodes que LDAP sont
  # possibles. Voir
  # <http://svnbook.red-bean.com/nightly/en/svn.serverconfig.httpd.html#svn.serverconfig.httpd.authz>
  AuthUserFile /dev/null
  AuthBasicProvider ldap
  AuthType Basic
  AuthName "Example & co Subversion Repository"
  AuthLDAPURL ldap://ldap.example.net/ou=People,dc=example,dc=net?uid?sub?(objectClass=*)

  # Le fichier où se trouvent les autorisations par répertoire
  AuthzSVNAccessFile /etc/apache2/subversion-access

  # On utilise "Satisfy any" car on veut permettre les accès anonymes aussi
  Satisfy any
  require valid-user

  </Location>

</VirtualHost>
```

Et ce fichier `/etc/apache2/subversion-access`, qui stocke les autorisations, que contient-il ? Il est documenté en `<http://svnbook.red-bean.com/en/1.5/svn.serverconfig.pathbasedauthz.html>`. Le principe est qu'on définit des groupes d'utilisateurs, puis on indique des permissions (droit de lire et d'écrire) à des répertoires du dépôt. Tous les répertoires ne sont évidemment pas listées systématiquement. Lorsqu'un répertoire n'est pas indiqué, Subversion utilise les permissions du parent, puis du grand-parent, etc. L'ordre des directives dans le fichier n'est donc pas vital mais la plupart des administrateurs mettent les répertoires du plus général au plus spécifique. Voici un exemple :

```
# Definition des groupes d'utilisateurs
[groups]
# La compagnie
staff = smith,jones,muller
# Des partenaires sur un projet
foobar = @staff,durand,li

# Par défaut : tout le monde peut lire, la compagnie peut lire et écrire
[/]
* = r
@staff = rw

[/Business]
# Inaccessible aux extérieurs. Comme les permissions sont normalement
# héritées du parent, il faut explicitement couper l'accès de "*" (tous).
* =
@staff = rw

[/FooBar]
# Un projet public, sur lequel nos partenaires travaillent
* = r
@foobar = rw
```

```
[/Secret]
# Projet très réservé
* =
smith = rw
```

À noter qu'il n'existe apparemment pas de moyen simple pour donner des droits à tous, sauf aux anonymes. Il faut lister tous les authentifiés explicitement (* inclus les anonymes).

Voici un accès à ce répertoire par un utilisateur anonyme, qui utilise le client Subversion en ligne de commande :

```
% svn co https://svn.example.net/
...
A    svn.example.net/FooBar/src/main.d
...
Checked out revision 7979.
```

Si, maintenant, je vais dans un des répertoires qui demandent une authentification :

```
% cd svn.example.net/FooBar/src
% emacs util.d
% svn add util.d
A         util.d
% svn commit
Authentication realm: <https://svn.example.net:443> Example & co Subversion Repository
Password for 'smith':
Adding         FooBar/src/util.d
Transmitting file data .
Committed revision 7980.
```

Apache demandera l'authentification, puisque l'anonyme n'a pas le droit d'écrire dans ce répertoire (notez que le client Subversion a affiché le domaine d'authentification, tel que configuré par `AuthName`).

L'un des inconvénients de l'utilisation d'Apache et de WebDAV est qu'il n'y a pas de rapport simple entre les actions Subversion ("*checkout*", "*commit*", etc) et les requêtes HTTP. L'examen du journal d'Apache n'est donc pas très utile. Un simple "*commit*" va produire une dizaine de requêtes dont :

```
88.189.152.187 - smith [22/Nov/2008:18:40:56 +0100] "PUT //!svn/wrk/8f2454bd-4a5c-0410-af3f-ad2595489424/FooBar/
88.189.152.187 - smith [22/Nov/2008:18:40:56 +0100] "MERGE /FooBar HTTP/1.1" 200 709 "-" "SVN/1.4.4 (r25188) neo
88.189.152.187 - smith [22/Nov/2008:18:40:56 +0100] "DELETE /!svn/act/8f2454bd-4a5c-0410-af3f-ad2595489424 HTTP/
```