Apache frontend for security

See Running Jenkins behind Apache for general information about running Jenkins on Apache. This page only covers the security aspect of it.

It is possible to use an apache in front of your tomcat instance that runs Jenkins. You will need to compile apache-2.2 with mod_proxy enabled. The example below shows an invocation of apache-2.2 configure script with parameters that enable mod_proxy, mod_proxy_ajp, LDAP and SSL.

```
[ root@buildhost# ]sudo ./configure --enable-proxy \
--enable-ldap \
--enable-vhost \
--enable-ssl \
--enable-suexec \
--enable-rewrite \
--enable-proxy-ajp \
--enable-authnz-ldap \
--enable-mods-shared=all \
--with-ssl \
--with-ldap \
--with-ldap-include=/usr/include/ \
--prefix=/opt/apache/httpd-2.2.6
```

Edit the httpd-vhosts.conf file that resides in ${APACHE_HOME}/conf/extras to make apache aware of your tomcat server.

The location of the file is different depending on how/where you install Apache from

Apache basic authentication

The example below shows a virtual host configuration for an apache that runs on the same machine as Jenkins. Jenkins here is configured to run an AJP connector on port 8102. This virtual host is also configured to rely on basic authentication (htpasswd) to protect certain resources, such as project(s) configuration, Jenkins management, and project(s) deletion. See the apache manual for examples of basic, and other, authentication scheme configuration.

```
<VirtualHost *:80>
    ServerAdmin your@email.address.com
    DocumentRoot "/opt/apache/httpd/htdocs"
    ServerName jenkins.yourdomain.com
    ErrorLog "logs/jenkins-error_log"

    ProxyPass /jenkins/ ajp://127.0.0.1:8102/jenkins/
    ProxyPassReverse /jenkins/ ajp://127.0.0.1:8102/jenkins/
    ProxyRequests Off
    <Location />
        Order allow,deny
        Allow from all
    </Location>
    <Location /jenkins/>
         AuthType basic
         AuthName "jenkins"
         AuthUserFile "/opt/apache/httpd/conf/.htpasswd"
         Require valid-user
 </Location>
</VirtualHost>
```

This approach is suitable if the access control need is simplistic (such as hiding Jenkins from everyone but a few people), but it tends to break down if you start doing more complex set up (such as making Jenkins visible to anonymous users but only allowing a few people to modify the settings.)

If you do access control in Apache, do not enable security in Jenkins, as those two things will interfere with each other.

Apache authentication against x509/SSL

Above requires the maintenance of an .htpassword file. If your organisation has issued certs (or relies on third party client certs such as issued by Thawte or Verisign) - below can be used to remove the need to maintain such a .htpassword file and not have any people 'secret's on the machine.

```
# Ensure that on top level SSL is active; along with
# authentication.
#
# SSLEngine       On
# SSLVerifyClient Require
# SSLVerifyDepth  3
# SSLCertificateFile    /etc/XXXX.pem
# SSLCertificateKeyFile /etc/XXXX.key

# 1) Expose CN and other variables to PHP, Perl, CGI
#    as SSL_CLIENT_S_DN, SSL_CLIENT_S_DN_CN, SSL_CLIENT_S_DN_emailAddress
#    (see http://httpd.apache.org/docs/2.2/mod/mod_ssl.html)
#
SSLOptions FakeBasicAuth

# 2) Next do a dirty trick to accept any users; yet rely on the
#    SSL SSLVerifyClient Require to make sure it is kosher.
#
#    Anyting in ZZZ will have the enivronment variable REMOTE_USER set.
<Location /jenkins/>
        AuthType Basic
        AuthName Jenkins
        AuthBasicProvider anon
        Anonymous *
        Require valid-user
</Location>
```

This will make request look to Jenkins just like those of the .htpassword sample of above.

Running Jenkins with AJP

Jenkins can be then run as the following to only listen to AJP connection on port 8102 without any HTTP listener. The 127.0.0.1 address also ensures that the external hosts cannot directly talk to Jenkins without going through Apache:

```
java -jar jenkins.war --httpPort=-1 --ajp13Port=8102 --ajp13ListenAddress=127.0.0.1
```

If you are running Jenkins inside a servlet container, refer to its documentation about how to prevent direct connection from outside to Jenkins. For example, in Tomcat, this is done by setting the **address** attribute in the tomcat connector definition. See http://tomcat.apache.org/tomcat-5.5-doc/config/ajp.html#Standard%20Implementation. For above localhost setting, use **address="127.0.0.1"**.