Version : **2023.01**.

Dernière mise-à-jour : 2023/12/04 09:22

SER305 - Sécurité du serveur Tomcat

Contenu du Module

- Configurer Apache
  - Installation de SSL
  - Configuration de SSL
  - Mise en place des paramètres de sécurité SSL
  - Tester Votre Configuration
- Apache en Frontal HTTPS
- Restrictions d'Accès
- Le Gestionnaire de Sécurité

Authentification, Autorisation et Cryptage

Authentification

Quand un client tente d'accéder à une ressource protégée d'un site ou d'une application web, le serveur renvoie un code HTTP **401**. A la réception de ce code, la navigateur affiche une boîte de dialogue d'authentification. Dans le cas où l'authentification n'aboutit pas, le serveur envoie un code HTTP **403** (Forbidden).

Pour mettre en œuvre ce mécanisme il existe quatre schémas d'authentification, dont les trois premiers sont :

- **BASIC** - l'utilisation de l'algorithme **Base64**,
- **DIGEST**, - l'utilisation d'un algorithme de hachage tel **SHA** ou **MD5**,
- **CLIENT-CERT** - l'utilisation de certificats HTTPS.

Les deux derniers schémas ci-dessus ne sont pas supportés par tous les navigateurs. Par conséquent, l'utilisation de l'authentification de base et HTTPS ensemble est plus courant.

Le quatrième schéma d'authentification est l'authentification par formulaire, **FORM**. Dans ce cas, le navigateur n'intervient pas car c'est le serveur qui sert un formulaire HTML au client.

Autorisation

Tomcat utilise un système **RBAC** (*Role Based Access Control*) pour l'autorisation. Dans ce cas, chaque utilisateur est attribué un ou plusieurs rôles dans le **registre d'authentification**.

Cryptage

Tomcat peut utiliser soit **SSL** soit **TLS** pour sécuriser le flux de données HTTP. Les technologies Java utilise les protocoles SSL et TLS dans la bibliothèque **JSSE** (*Java Secure Socket Extention*).

La Sécurité sous Tomcat

Configuration

La configuration de la sécurité se fait dans le fichier **web.xml** de l'application concernée. Consultez le fichier **/usr/tomcat8/webapps/examples/WEB-INF/web.xml**. A la fin de celui-ci, trouvez l'élément **<security-constraint>** :

```
...
   <security-constraint>
      <display-name>Example Security Constraint - part 1</display-name>
      <web-resource-collection>
         <web-resource-name>Protected Area - Allow methods</web-resource-name>
         <!-- Define the context-relative URL(s) to be protected -->
         <url-pattern>/jsp/security/protected/*</url-pattern>
         <!-- If you list http methods, only those methods are protected so -->
         <!-- the constraint below ensures all other methods are denied     -->
         <http-method>DELETE</http-method>
         <http-method>GET</http-method>
         <http-method>POST</http-method>
         <http-method>PUT</http-method>
      </web-resource-collection>
      <auth-constraint>
```

```
        <!-- Anyone with one of the listed roles may access this area -->
        <role-name>tomcat</role-name>
        <role-name>role1</role-name>
    </auth-constraint>
  </security-constraint>
 <security-constraint>
    <display-name>Example Security Constraint - part 2</display-name>
    <web-resource-collection>
       <web-resource-name>Protected Area - Deny methods</web-resource-name>
       <!-- Define the context-relative URL(s) to be protected -->
       <url-pattern>/jsp/security/protected/*</url-pattern>
       <http-method-omission>DELETE</http-method-omission>
       <http-method-omission>GET</http-method-omission>
       <http-method-omission>POST</http-method-omission>
       <http-method-omission>PUT</http-method-omission>
    </web-resource-collection>
    <!-- An empty auth constraint denies access -->
    <auth-constraint />
  </security-constraint>

  <!-- Default login configuration uses form-based authentication -->
  <login-config>
    <auth-method>FORM</auth-method>
    <realm-name>Example Form-Based Authentication Area</realm-name>
    <form-login-config>
      <form-login-page>/jsp/security/protected/login.jsp</form-login-page>
      <form-error-page>/jsp/security/protected/error.jsp</form-error-page>
    </form-login-config>
  </login-config>

  <!-- Security roles referenced by this web application -->
  <security-role>
    <role-name>role1</role-name>
  </security-role>
```

```
    <security-role>
      <role-name>tomcat</role-name>
    </security-role>
...
```

L'élément <security-constraint> contient d'autres éléments dont les plus importants sont :

Elément	**Description**
<web-resource-collection></web-resource-collection>	Contient les ressources à protéger
<auth-constraint></auth-constraint>	Indique les rôles qui auront accès au ressources protégées

L'élément <Login-conf> contient d'autres éléments dont les plus importants sont :

Elément	**Description**
<auth-method></auth-method>	Vaut BASIC, DIGEST, CLIENT-CERT ou FORM
<form-login-page></form-login-page>	Indique la page contenant le formulaire
<form-error-page></form-error-page>	Indique la page d'erreur envoyée au client en cas d'échec d'authentification

L'élément <Security-role> doit contenir un élément **<role-name>** pour chaque rôle à déclarer.

Pour utiliser l'authentification par formulaire, celui-ci doit :

- être posté à destination du servlet **j_security_check**,
- posséder le champ **j_username** pour recevoir le nom de l'utilisateur,
- posséder le champ **j_password** pour recevoir le mote de passe de l'utilisateur.

Realms

L'accès au **registre d'authentification** est obtenu en utilisant des **Realms**. Tomcat peut utiliser les sept Realms suivants :

- User Database Realm,
- JDBC Realm,
- DataSource Realm,

- JNDI Realm,
- JAAS Realm,
- Combined Realm,
- LockOut Realm.

User Database Realm

Ce type de Realm utilise la classe Java **org.apache.catalina.realm.UserDatabaseRealm**. Les informations sont stockées dans un fichier XML qui est par défaut **$CATALINA_HOME/conf/tomcat-users.xml** :

```
<?xml version='1.0' encoding='utf-8'?>
<tomcat-users xmlns="http://tomcat.apache.org/xml"
              xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
              xsi:schemaLocation="http://tomcat.apache.org/xml tomcat-users.xsd"
              version="1.0">
  <role rolename="tomcat"/>
  <role rolename="role1"/>
  <role rolename="manager-script"/>
  <user username="tomcat" password="tomcat" roles="tomcat"/>
  <user username="both" password="tomcat" roles="tomcat,role1"/>
  <user username="role1" password="tomcat" roles="role1"/>
  <user username="admin" password="fenestros" roles="manager-script"/>
</tomcat-users>
```

La configuration de ce Realm se trouve dans le fichier **$CATALINA_HOME/conf/server.xml** dans les éléments <GlobalNamingResources> et <Engine> :

```
  <GlobalNamingResources>
    <!-- Editable user database that can also be used by
         UserDatabaseRealm to authenticate users
    -->
    <Resource name="UserDatabase" auth="Container"
              type="org.apache.catalina.UserDatabase"
```

```
              description="User database that can be updated and saved"
              factory="org.apache.catalina.users.MemoryUserDatabaseFactory"
              pathname="conf/tomcat-users.xml" />
  </GlobalNamingResources>
  ...
    <Engine name="Catalina" defaultHost="localhost">
  ...
        <Realm className="org.apache.catalina.realm.UserDatabaseRealm"
               resourceName="UserDatabase"/>
      </Realm>
  ...
```

Dans le cas ci-dessus, les mots de passe sont en clair dans le fichier $CATALINA_HOME/conf/tomcat-users.xml. Il est cependant possible de les cryptés grâce à la classe **javax.security.MessageDigest** en utilisant soit l'algorithme **SHA** soit l'algorithme **MD5** :

```
[root@centos7 ~]# cd $CATALINA_HOME/bin
[root@centos7 bin]# ./digest.sh -a sha fenestros
fenestros:75affcc6adf7ec7cd21f6479b8fb87b89a550b2602e613715d57d862b15c7c17$1$015b4b320315c87572918dbcc08b65a240d0
a750
```

Il est ensuite nécessaire d'éditer le fichier **$CATALINA_HOME/conf/tomcat-users.xml** en remplacant le mot de passe en clair “fenestros” avec le mot de passe crypté :

```
[root@centos7 bin]# vi $CATALINA_HOME/conf/tomcat-users.xml
[root@centos7 bin]# cat $CATALINA_HOME/conf/tomcat-users.xml
<?xml version='1.0' encoding='utf-8'?>
<tomcat-users xmlns="http://tomcat.apache.org/xml"
              xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
              xsi:schemaLocation="http://tomcat.apache.org/xml tomcat-users.xsd"
              version="1.0">
  <role rolename="tomcat"/>
  <role rolename="role1"/>
  <role rolename="manager-script"/>
  <user username="tomcat" password="tomcat" roles="tomcat"/>
```

```
  <user username="both" password="tomcat" roles="tomcat,role1"/>
  <user username="role1" password="tomcat" roles="role1"/>
  <user username="admin"
password="75affcc6adf7ec7cd21f6479b8fb87b89a550b2602e613715d57d862b15c7c17$1$015b4b320315c87572918dbcc08b65a240d0
a750" roles="manager-script"/>
</tomcat-users>
```

> **Important** : NE COPIEZ PAS simplement l'exemple du fichier ci-dessus. MODIFIEZ le fichier en remplaçant le mot de passe fenestros avec le mot de passe crypté que VOUS obtenez en exécutant la commande **./digest.sh -a sha fenestros**.

Dernièrement il faut éditer le fichier **$CATALINA_HOME/conf/server.xml**

```
...
    <Engine name="Catalina" defaultHost="localhost">
  ...
        <Realm className="org.apache.catalina.realm.UserDatabaseRealm"
               resourceName="UserDatabase" digest="sha" />
      </Realm>
...
```

Redémarrez le serveur Tomcat :

```
[root@centos7 bin]# ./shutdown.sh
Using CATALINA_BASE:   /usr/tomcat8
Using CATALINA_HOME:   /usr/tomcat8
Using CATALINA_TMPDIR: /usr/tomcat8/temp
Using JRE_HOME:        /usr/lib/jvm/jre-1.8.0-openjdk-1.8.0.232.b09-0.el7_7.x86_64
Using CLASSPATH:       /usr/tomcat8/bin/bootstrap.jar:/usr/tomcat8/bin/tomcat-juli.jar
[root@centos7 bin]# ./startup.sh
Using CATALINA_BASE:   /usr/tomcat8
Using CATALINA_HOME:   /usr/tomcat8
```

```
Using CATALINA_TMPDIR: /usr/tomcat8/temp
Using JRE_HOME:        /usr/lib/jvm/jre-1.8.0-openjdk-1.8.0.232.b09-0.el7_7.x86_64
Using CLASSPATH:       /usr/tomcat8/bin/bootstrap.jar:/usr/tomcat8/bin/tomcat-juli.jar
Tomcat started.
```

Testez votre connexion :

```
[root@centos7 bin]# lynx --dump -auth admin:fenestros "http://www.i2tch.loc:8080/manager/text/serverinfo"
OK - Server info
Tomcat Version: Apache Tomcat/8.0.36
OS Name: Linux
OS Version: 3.10.0-1062.4.1.el7.x86_64
OS Architecture: amd64
JVM Version: 1.8.0_232-b09
JVM Vendor: Oracle Corporation
```

JDBC Realm

Ce type de Realm utilise la classe Java **org.apache.catalina.realm.JDBCRealm**. Les informations sont stockées dans une base de données dont le script MySQL est le suivant :

```
CREATE DATABASE `auth_tomcat`;
USE `auth_tomcat`;
CREATE TABLE `auth_tomcat`.`users` (
        `nom_user` varchar(45) NOT NULL,
        `mdp_user` varchar(45) NOT NULL,
        PRIMARY KEY (`nom_user`)
) ENGINE=InnoDB DEFAULT CHARSET=latin1;
CREATE TABLE `auth_tomcat`.`roles` (
        `nom_user` varchar(45) NOT NULL,
        `nom_role` varchar(45) NOT NULL,
        PRIMARY KEY (`nom_user`)
```

```
) ENGINE=InnoDB DEFAULT CHARSET=latin1;
```

Par exemple :

```
[root@centos7 bin]# cd ~
[root@centos7 ~]# vi auth_tomcat
[root@centos7 ~]# cat auth_tomcat
CREATE DATABASE `auth_tomcat`;
USE `auth_tomcat`;
CREATE TABLE `auth_tomcat`.`users` (
        `nom_user` varchar(45) NOT NULL,
        `mdp_user` varchar(45) NOT NULL,
        PRIMARY KEY (`nom_user`)
) ENGINE=InnoDB DEFAULT CHARSET=latin1;
CREATE TABLE `auth_tomcat`.`roles` (
        `nom_user` varchar(45) NOT NULL,
        `nom_role` varchar(45) NOT NULL,
        PRIMARY KEY (`nom_user`)
) ENGINE=InnoDB DEFAULT CHARSET=latin1;
```

Créez donc la basee de données **auth_tomcat** :

```
[root@centos7 ~]# mysql -u root -p < auth_tomcat
Enter password: fenestros
[root@centos7 ~]#
```

Connectez-vous au serveur MariaDB et créez l'utilisateur **admin1** ayant un mot de passe **fenestros** et un rôle **manager-script** :

```
[root@centos7 ~]# mysql -u root -p
Enter password: fenestros
Welcome to the MariaDB monitor.  Commands end with ; or \g.
Your MariaDB connection id is 12
Server version: 5.5.47-MariaDB MariaDB Server
```

```
Copyright (c) 2000, 2015, Oracle, MariaDB Corporation Ab and others.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

MariaDB [(none)]> SHOW databases;
+--------------------+
| Database           |
+--------------------+
| information_schema |
| auth_tomcat        |
| mysql              |
| performance_schema |
| test               |
| tomcat             |
+--------------------+
6 rows in set (0.01 sec)

MariaDB [(none)]> USE auth_tomcat;
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Database changed
MariaDB [auth_tomcat]> INSERT INTO `auth_tomcat`.`users` VALUES('admin1','fenestros');
Query OK, 1 row affected (0.01 sec)

MariaDB [auth_tomcat]> INSERT INTO `auth_tomcat`.`roles` VALUES('admin1','manager-script');
Query OK, 1 row affected (0.02 sec)

MariaDB [auth_tomcat]> GRANT SELECT ON auth_tomcat.* TO 'tomcat'@'localhost' IDENTIFIED BY 'tomcat';
Query OK, 0 rows affected (0.10 sec)

MariaDB [auth_tomcat]> FLUSH PRIVILEGES;
Query OK, 0 rows affected (0.02 sec)
```

```
MariaDB [auth_tomcat]> SET PASSWORD FOR 'tomcat'@'localhost' = PASSWORD('secret');
Query OK, 0 rows affected (0.02 sec)

MariaDB [auth_tomcat]> exit
Bye
```

Modifiez le fichier **$CATALINA_HOME/conf/server.xml** en mettant en commentaires le Realm **In-Memory** et en ajoutant la section suivante :

```
...
        <!-- <Realm className="org.apache.catalina.realm.UserDatabaseRealm"
               resourceName="UserDatabase" digest="sha" /> -->

      <Realm  className="org.apache.catalina.realm.JDBCRealm"
             driverName="com.mysql.jdbc.Driver"
          connectionURL="jdbc:mysql://localhost:3306/auth_tomcat"
         connectionName="tomcat" connectionPassword="secret"
              userTable="users" userNameCol="nom_user" userCredCol="mdp_user"
          userRoleTable="roles" roleNameCol="nom_role" />

      </Realm>
...
```

Redémarrez le serveur Tomcat :

```
[root@centos7 ~]# cd $CATALINA_HOME/bin
[root@centos7 bin]# ./shutdown.sh
Using CATALINA_BASE:   /usr/tomcat8
Using CATALINA_HOME:   /usr/tomcat8
Using CATALINA_TMPDIR: /usr/tomcat8/temp
Using JRE_HOME:        /usr/lib/jvm/jre-1.8.0-openjdk-1.8.0.232.b09-0.el7_7.x86_64
Using CLASSPATH:       /usr/tomcat8/bin/bootstrap.jar:/usr/tomcat8/bin/tomcat-juli.jar
[root@centos7 bin]# ./startup.sh
Using CATALINA_BASE:   /usr/tomcat8
Using CATALINA_HOME:   /usr/tomcat8
```

```
Using CATALINA_TMPDIR: /usr/tomcat8/temp
Using JRE_HOME:        /usr/lib/jvm/jre-1.8.0-openjdk-1.8.0.232.b09-0.el7_7.x86_64
Using CLASSPATH:       /usr/tomcat8/bin/bootstrap.jar:/usr/tomcat8/bin/tomcat-juli.jar
Tomcat started.
```

Vérifiez maintenant que vous pouvez vous connecter avec le compte **admin1** :

```
[root@centos7 bin]# lynx --dump -auth admin1:fenestros "http://www.i2tch.loc:8080/manager/text/serverinfo"
OK - Server info
Tomcat Version: Apache Tomcat/8.0.36
OS Name: Linux
OS Version: 3.10.0-1062.4.1.el7.x86_64
OS Architecture: amd64
JVM Version: 1.8.0_232-b09
JVM Vendor: Oracle Corporation
```

DataSource Realm

Ce type de Realm utilise la classe Java **org.apache.catalina.realm.DataSourceRealm**. Les informations sont stockées dans une base de données identique au cas du Realm JDBCRealm. La différence principale du Realm DataSource par rapport au Realm JDBC est que le premier peut utiliser un **pool** de connexions augmentant ainsi la performance.

Créez d'abord l'utilisateur **admin2** dans la base de données **auth_tomcat** :

```
[root@centos7 bin]# mysql -u root -p
Enter password:
Welcome to the MariaDB monitor.  Commands end with ; or \g.
Your MariaDB connection id is 2
Server version: 5.5.47-MariaDB MariaDB Server

Copyright (c) 2000, 2015, Oracle, MariaDB Corporation Ab and others.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.
```

```
MariaDB [(none)]> USE auth_tomcat;
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Database changed
MariaDB [auth_tomcat]> INSERT INTO `auth_tomcat`.`users` VALUES('admin2','fenestros');
Query OK, 1 row affected (0.01 sec)

MariaDB [auth_tomcat]> INSERT INTO `auth_tomcat`.`roles` VALUES('admin2','manager-script');
Query OK, 1 row affected (0.00 sec)

MariaDB [auth_tomcat]> exit
Bye
[root@centos7 bin]#
```

Modifiez ensuite le fichier **$CATALINA_HOME/conf/server.xml** en y ajoutant un élément **<Resource name="jdbc/AuthTomcat" ...>** dans l'élément **<GlobalNamingResource>** :

```
...
  <GlobalNamingResources>
    <!-- Editable user database that can also be used by
         UserDatabaseRealm to authenticate users
    -->
    <Resource name="UserDatabase" auth="Container"
              type="org.apache.catalina.UserDatabase"
              description="User database that can be updated and saved"
              factory="org.apache.catalina.users.MemoryUserDatabaseFactory"
              pathname="conf/tomcat-users.xml" />

    <Resource name="jdbc/AuthTomcat" auth="Container"
              type="javax.sql.DataSource"
          driverName="com.mysql.jdbc.Driver"
                 url="jdbc:mysql://localhost:3306/auth_tomcat"
            username="tomcat"
```

```
          password="secret" />

  </GlobalNamingResources>
...
```

Ensuite, éditez l'élément Realm précédemment créé ainsi :

```
...
        <!-- <Realm className="org.apache.catalina.realm.UserDatabaseRealm"
               resourceName="UserDatabase" digest="sha" /> -->

      <!-- <Realm  className="org.apache.catalina.realm.JDBCRealm"
             driverName="com.mysql.jdbc.Driver"
          connectionURL="jdbc:mysql://localhost:3306/auth_tomcat"
         connectionName="tomcat" connectionPassword="secret"
              userTable="users" userNameCol="nom_user" userCredCol="mdp_user"
          userRoleTable="roles" roleNameCol="nom_role" />

      </Realm> -->

      <Realm  className="org.apache.catalina.realm.DataSourceRealm"
         dataSourceName="jdbc/AuthTomcat"
              userTable="users" userNameCol="nom_user" userCredCol="mdp_user"
          userRoleTable="roles" roleNameCol="nom_role" />
    </Realm>
...
```

Redémarrez le serveur Tomcat :

```
[root@centos7 bin]# ./shutdown.sh
Using CATALINA_BASE:   /usr/tomcat8
Using CATALINA_HOME:   /usr/tomcat8
Using CATALINA_TMPDIR: /usr/tomcat8/temp
Using JRE_HOME:        /usr/lib/jvm/jre-1.8.0-openjdk-1.8.0.232.b09-0.el7_7.x86_64
```

```
Using CLASSPATH:       /usr/tomcat8/bin/bootstrap.jar:/usr/tomcat8/bin/tomcat-juli.jar
[root@centos7 bin]# ./startup.sh
Using CATALINA_BASE:   /usr/tomcat8
Using CATALINA_HOME:   /usr/tomcat8
Using CATALINA_TMPDIR: /usr/tomcat8/temp
Using JRE_HOME:        /usr/lib/jvm/jre-1.8.0-openjdk-1.8.0.232.b09-0.el7_7.x86_64
Using CLASSPATH:       /usr/tomcat8/bin/bootstrap.jar:/usr/tomcat8/bin/tomcat-juli.jar
Tomcat started.
```

Vérifiez maintenant que vous pouvez vous connecter avec le compte **admin2** :

```
[root@centos7 bin]# lynx --dump -auth admin2:fenestros "http://www.i2tch.loc:8080/manager/text/serverinfo"
OK - Server info
Tomcat Version: Apache Tomcat/8.0.36
OS Name: Linux
OS Version: 3.10.0-1062.4.1.el7.x86_64
OS Architecture: amd64
JVM Version: 1.8.0_232-b09
JVM Vendor: Oracle Corporation
```

JNDI Realm

Ce type de Realm utilise la classe Java **org.apache.catalina.realm.JNDIRealm**. Les informations sont stockées dans une base de données LDAP.

Le format LDIF

Les fichiers au format LDIF (**LD**AP **I**nterchange **F**ormat) sont utilisés lors de modifications de masse sur une base LDAP. Les fichiers LDIF sont traités dans un ordre séquentielle.

Le fichier LDIF est un fichier texte qui peut comprendre :

- des descriptions d'entrées de l'annuaire,
- des valeurs d'attribut pour les entrées de l'annuaire,
- des instructions de traitements pour le serveur.

Un fichier LDIF peut comporter des commentaires à l'aide du caractère **#**. Chaque enregistrement doit être séparé du précédent par une ligne blanche et il ne peut pas avoir deux lignes blanches consécutives.

Les attributs peuvent être sur plusieurs lignes. Dans ce cas les lignes supplémentaires commencent par un blanc. Par exemple :

```
dn: o=fenestros.loc
objectClass: dcObject
objectClass: organization
dc: fenestros
o: fenestros.loc
description: Exemple

dn: ou=utilisateurs,o=fenestros.loc
objectClass: organizationalUnit
objectClass: top
ou: utilisateurs

dn: cn=admin3,ou=utilisateurs,o=fenestros.loc
objectClass: person
objectClass: top
cn: admin3
sn: admin3
userPassword: fenestros

dn: ou=roles,o=fenestros.loc
objectClass: organizationalUnit
objectClass: top
ou: roles

dn: cn=manager-script,ou=roles,o=fenestros.loc
```

```
objectClass: groupOfUniqueNames
objectClass: top
cn: manager-script
uniqueMember: cn=admin3,ou=utilisateurs,o=fenestros.loc
```

La commande ldapadd

Afin de pouvoir utiliser notre fichier LDIF, il est nécessaire de faire appel au client **ldapadd**. Cet utilitaire prend un ou plusieurs options :

```
[root@centos7 bin]# ldapadd --help
ldapadd: invalid option -- '-'
ldapadd: unrecognized option --
Add or modify entries from an LDAP server

usage: ldapadd [options]
    The list of desired operations are read from stdin or from the file
    specified by "-f file".
Add or modify options:
  -a         add values (default)
  -c         continuous operation mode (do not stop on errors)
  -E [!]ext=extparam    modify extensions (! indicate s criticality)
  -f file    read operations from `file'
  -M         enable Manage DSA IT control (-MM to make critical)
  -P version protocol version (default: 3)
  -S file    write skipped modifications to `file'
Common options:
  -d level   set LDAP debugging level to `level'
  -D binddn  bind DN
  -e [!]<ext>[=<extparam>] general extensions (! indicates criticality)
             [!]assert=<filter>     (RFC 4528; a RFC 4515 Filter string)
             [!]authzid=<authzid>   (RFC 4370; "dn:<dn>" or "u:<user>")
             [!]chaining[=<resolveBehavior>[/<continuationBehavior>]]
                    one of "chainingPreferred", "chainingRequired",
```

```
                        "referralsPreferred", "referralsRequired"
            [!]manageDSAit         (RFC 3296)
            [!]noop
            ppolicy
            [!]postread[=<attrs>]  (RFC 4527; comma-separated attr list)
            [!]preread[=<attrs>]   (RFC 4527; comma-separated attr list)
            [!]relax
            [!]sessiontracking
            abandon, cancel, ignore (SIGINT sends abandon/cancel,
            or ignores response; if critical, doesn't wait for SIGINT.
            not really controls)
  -h host    LDAP server
  -H URI     LDAP Uniform Resource Identifier(s)
  -I         use SASL Interactive mode
  -n         show what would be done but don't actually do it
  -N         do not use reverse DNS to canonicalize SASL host name
  -O props   SASL security properties
  -o <opt>[=<optparam>] general options
             nettimeout=<timeout> (in seconds, or "none" or "max")
             ldif-wrap=<width> (in columns, or "no" for no wrapping)
  -p port    port on LDAP server
  -Q         use SASL Quiet mode
  -R realm   SASL realm
  -U authcid SASL authentication identity
  -v         run in verbose mode (diagnostics to standard output)
  -V         print version info (-VV only)
  -w passwd  bind password (for simple authentication)
  -W         prompt for bind password
  -x         Simple authentication
  -X authzid SASL authorization identity ("dn:<dn>" or "u:<user>")
  -y file    Read password from file
  -Y mech    SASL mechanism
  -Z         Start TLS request (-ZZ to require successful response)
```

Créez maintenant le fichier **/etc/openldap/slapd.conf** :

```
include     /etc/openldap/schema/corba.schema
include     /etc/openldap/schema/core.schema
include     /etc/openldap/schema/cosine.schema
include     /etc/openldap/schema/duaconf.schema
include     /etc/openldap/schema/dyngroup.schema
include     /etc/openldap/schema/inetorgperson.schema
include     /etc/openldap/schema/java.schema
include     /etc/openldap/schema/misc.schema
include     /etc/openldap/schema/nis.schema
include     /etc/openldap/schema/openldap.schema
include     /etc/openldap/schema/ppolicy.schema
include     /etc/openldap/schema/collective.schema
allow bind_v2
pidfile     /var/run/openldap/slapd.pid
argsfile    /var/run/openldap/slapd.args
TLSCACertificatePath /etc/openldap/certs
TLSCertificateFile "\"OpenLDAP Server\""
TLSCertificateKeyFile /etc/openldap/certs/password
database config
access to *
    by dn.exact="gidNumber=0+uidNumber=0,cn=peercred,cn=external,cn=auth" manage
    by * none
database monitor
access to *
    by dn.exact="gidNumber=0+uidNumber=0,cn=peercred,cn=external,cn=auth" read
        by dn.exact="cn=Manager,dc=my-domain,dc=com" read
        by * none
database    bdb
suffix          "o=fenestros.loc"
checkpoint  1024 15
rootdn          "cn=Manager,o=fenestros.loc"
rootpw      fenestros
```

```
directory   /var/lib/ldap
index objectClass                       eq,pres
index ou,cn,mail,surname,givenname      eq,pres,sub
index uidNumber,gidNumber,loginShell    eq,pres
index uid,memberUid                     eq,pres,sub
index nisMapName,nisMapEntry            eq,pres,sub
```

Nettoyez les anciens fichiers de configuration et fichiers de données :

```
[root@centos7 ~]# rm -Rf /etc/openldap/slapd.d/*
[root@centos7 ~]# rm -f /var/lib/ldap/alock
[root@centos7 ~]# rm -f /var/lib/ldap/__db.00?
```

Copiez le fichier /usr/share/openldap-servers/DB_CONFIG.example vers **/var/lib/ldap/DB_CONFIG** :

```
[root@centos7 ~]# cp /usr/share/openldap-servers/DB_CONFIG.example /var/lib/ldap/DB_CONFIG
```

Initialisez ensuite l'arborescence dans **/etc/openldap/slapd.d** :

```
[root@centos7 ~]# slaptest -f /etc/openldap/slapd.conf -F /etc/openldap/slapd.d
config file testing succeeded
```

Vérifiez que l'arborescence initiale soit créée :

```
[root@centos7 ~]# ls -l /etc/openldap/slapd.d
total 8
drwxr-x--- 3 root root 4096 Nov  5 13:20 cn=config
-rw------- 1 root root 1258 Nov  5 13:20 cn=config.ldif
```

Modifiez le propriétaire, le groupe ainsi que le droits du répertoire **/etc/openldap/slapd.d** :

```
[root@centos7 ~]# chown -R ldap:ldap /etc/openldap/slapd.d
[root@centos7 ~]# chmod -R u+rwX /etc/openldap/slapd.d
```

Modifiez le propriétaire et le groupe répertoire **/var/lib/ldap/** ainsi que le fichier **/etc/openldap/slapd.conf** :

```
[root@centos7 ~]# chown -R ldap:ldap /var/lib/ldap/ /etc/openldap/slapd.conf
```

Redémarrez le service slapd :

```
[root@centos7 bin]# systemctl restart slapd.service
```

Créez maintenant notre fichier LDIF dans le répertoire /tmp :

```
[root@centos7 bin]# cd /tmp
[root@centos7 tmp]# vi setup.ldif
[root@centos7 tmp]# cat setup.ldif
dn: o=fenestros.loc
objectClass: dcObject
objectClass: organization
dc: fenestros
o: fenestros.loc
description: Exemple

dn: ou=utilisateurs,o=fenestros.loc
objectClass: organizationalUnit
objectClass: top
ou: utilisateurs

dn: cn=admin3,ou=utilisateurs,o=fenestros.loc
objectClass: person
objectClass: top
cn: admin3
sn: admin3
userPassword: fenestros

dn: ou=roles,o=fenestros.loc
objectClass: organizationalUnit
```

```
objectClass: top
ou: roles

dn: cn=manager-script,ou=roles,o=fenestros.loc
objectClass: groupOfUniqueNames
objectClass: top
cn: manager-script
uniqueMember: cn=admin3,ou=utilisateurs,o=fenestros.loc
```

Il convient maintenant d'utiliser la commande ldapadd afin d'injecter le contenu du fichier setup.ldif dans notre base :

```
[root@centos7 tmp]# ldapadd -f setup.ldif -x -D "cn=Manager,o=fenestros.loc" -w fenestros
adding new entry "o=fenestros.loc"

adding new entry "ou=utilisateurs,o=fenestros.loc"

adding new entry "cn=admin3,ou=utilisateurs,o=fenestros.loc"

adding new entry "ou=roles,o=fenestros.loc"

adding new entry "cn=manager-script,ou=roles,o=fenestros.loc"
```

L'arborescence LDAP est la suivante :

```
localhost
        |
    o=fenestros.loc
        |
        ou=roles
        |    |
        |    cn=manager-script
        |
        ou=users
            |
```

```
            cn=admin3
```

Ajoutez ensuite la section suivante au fichier **$CATALINA_HOME/conf/server.xml** en mettant en commentaires le <Realm> précédent :

```
...
<!-- <Realm className="org.apache.catalina.realm.UserDatabaseRealm"
               resourceName="UserDatabase" digest="sha" /> -->

<!--  <Realm  className="org.apache.catalina.realm.JDBCRealm"
             driverName="com.mysql.jdbc.Driver"
          connectionURL="jdbc:mysql://localhost:3306/auth_tomcat"
         connectionName="tomcat" connectionPassword="secret"
              userTable="users" userNameCol="nom_user" userCredCol="mdp_user"
          userRoleTable="roles" roleNameCol="nom_role" />

      </Realm> -->

<!--   <Realm  className="org.apache.catalina.realm.DataSourceRealm"
         dataSourceName="jdbc/AuthTomcat"
              userTable="users" userNameCol="nom_user" userCredCol="mdp_user"
          userRoleTable="roles" roleNameCol="nom_role" />
    </Realm> -->

      <Realm  className="org.apache.catalina.realm.JNDIRealm"
          connectionURL="ldap://localhost:389"
         connectionName="cn=Manager,o=fenestros.loc"
     connectionPassword="fenestros"
               roleBase="ou=roles,o=fenestros.loc"
           roleName="cn"
         roleSearch="(uniqueMember={0})"
           userPassword="userPassword"
            userPattern="cn={0},ou=utilisateurs,o=fenestros.loc" />
    </Realm>
```

```
      <Host name="localhost"  appBase="webapps"
            unpackWARs="true" autoDeploy="true">
...
```

Redémarrez ensuite le service tomcat :

```
[root@centos7 tmp]# cd $CATALINA_HOME/bin
[root@centos7 bin]# ./shutdown.sh
Using CATALINA_BASE:   /usr/tomcat8
Using CATALINA_HOME:   /usr/tomcat8
Using CATALINA_TMPDIR: /usr/tomcat8/temp
Using JRE_HOME:        /usr/lib/jvm/jre-1.8.0-openjdk-1.8.0.232.b09-0.el7_7.x86_64
Using CLASSPATH:       /usr/tomcat8/bin/bootstrap.jar:/usr/tomcat8/bin/tomcat-juli.jar
[root@centos7 bin]# ./startup.sh
Using CATALINA_BASE:   /usr/tomcat8
Using CATALINA_HOME:   /usr/tomcat8
Using CATALINA_TMPDIR: /usr/tomcat8/temp
Using JRE_HOME:        /usr/lib/jvm/jre-1.8.0-openjdk-1.8.0.232.b09-0.el7_7.x86_64
Using CLASSPATH:       /usr/tomcat8/bin/bootstrap.jar:/usr/tomcat8/bin/tomcat-juli.jar
Tomcat started.
```

Vérifiez maintenant que vous pouvez vous connecter avec le compte **admin3** :

```
[root@centos7 bin]# lynx --dump -auth admin3:fenestros "http://www.i2tch.loc:8080/manager/text/serverinfo"
OK - Server info
Tomcat Version: Apache Tomcat/8.0.36
OS Name: Linux
OS Version: 3.10.0-1062.4.1.el7.x86_64
OS Architecture: amd64
JVM Version: 1.8.0_232-b09
JVM Vendor: Oracle Corporation
```

JAAS Realm

Le Realm JAAS est utilisé pour authentifier un utilisateur contre n'importe quel registre de stockage d'informations en développant un module d'authentification adéquat pour le registre concerné.

L'étude du développement d'un tel module dépasse le cadre de cette formation.

Combined Realm

Ce type de Realm utilise la classe Java **org.apache.catalina.realm.CombinedRealm**. Il permet de chaîner plusieurs Realms pour l'authentification afin de fournir une solution de disponibilité sur l'authentification.

Ce Realm ne contient qu'un seul attribut : **className**.

Voici un **exemple** de la section Realm du fichier $CATALINA_HOME/conf/server.xml :

```
    <Realm className="org.apache.catalina.realm.CombinedRealm">
            <Realm className="org.apache.catalina.realm.UserDatabaseRealm"
            resourceName="UserDatabase_1"/>
            <Realm className="org.apache.catalina.realm.UserDatabaseRealm"
            resourceName="UserDatabase_2"/>
    </Realm>
```

Important : L'authentification d'un seul des ces sous-Realms est suffisante pour autoriser l'utilisateur.

LockOut Realm

Ce type de Realm utilise la classe Java **org.apache.catalina.realm.LockOutRealm**. Il permet d'appliquer des des règles de blocage de comptes pour

les sous-Realms qu'il englobe.

Ce Realm contient trois attributs :

- **className**,
- **failureCount**,
  - Spécifie le nombre de tentatives echouées de connexion avant un blocage. Par défaut la valeur est de **5**.
- **lockOutTime**,
  - Spécifie le nombre de secondes que le compte est bloqué. Par défaut la valeur est de **300**.

Voici un **exemple** de la section Realm du fichier $CATALINA_HOME/conf/server.xml :

```
    <Realm className="org.apache.catalina.realm.LockOutRealm">
            <Realm className="org.apache.catalina.realm.UserDatabaseRealm"
            resourceName="UserDatabase"/>
    </Realm>
```

Tomcat et le SSO

Le comportement par défaut de Tomcat est de demander une authentification par application.

Dans le cas où on souhaite mettre en place un **Single Sign-On**, il convient d'utiliser un élément <Valve>.

La configuration est la suivante :

```
<Host name ="localhost" ...>
  ...
  <Valve className="org.apache.catalina.authenticator.SingleSignOn" debug="0" />
  ...
```

Tomcat et le SSL

Présentation de SSL

SSL (*Secure Sockets Layers*) est utilisé pour sécuriser des transactions effectuées sur le Web et a été mis au point par :

- Netscape
- MasterCard
- Bank of America
- MCI
- Silicon Graphics

SSL est indépendant du protocole utilisé et agit en tant que couche supplémentaire entre la couche Application et la couche Transport. Il peut être utilisé avec :

- HTTP
- FTP
- POP
- IMAP

Fonctionnement de SSL

Le fonctionnement de SSL suit la procédure suivante :

- Le navigateur demande une page web sécurisée en https,
- Le serveur web émet sa clé publique et son certificat,
- Le navigateur vérifie que le certificat a été émis par une autorité fiable, qu'il est valide et qu'il fait référence au site consulté,
- Le navigateur utilise la clé publique du serveur pour chiffrer une clé symétrique aléatoire, une clé de session, et l'envoie au serveur avec l'URL demandé ainsi que des données HTTP chiffrées,
- Le serveur déchiffre la clé symétrique avec sa clé privée et l'utilise pour récupérer l'URL demandé et les données HTTP,
- Le serveur renvoie le document référencé par l'URL ainsi que les données HTTP chiffrées avec la clé symétrique,
- Le navigateur déchiffre le tout avec la clé symétrique et affiche les informations.

Quand on parle de **SSL**, on parle de **cryptologie**.

Configurer Tomcat

Pour utiliser SSL avec Tomcat, il est nécessaire d'avoir un répertoire pour stocker le fichier **.keystore**. Ce fichier doit contenir le certificat du serveur.

Commencez donc par créer ce répertoire :

```
[root@centos7 bin]# mkdir $CATALINA_HOME/security
```

Pour générer le certificat, il faut d'abord créer une clef privée. Cette clef est créée par la commande **keytool** :

```
[root@centos7 bin]# keytool -genkey -alias tomcat -keyalg RSA -keystore
$CATALINA_HOME/security/www_i2tch_loc.keystore
Enter keystore password: fenestros
Re-enter new password: fenestros
What is your first and last name?
  [Unknown]:  HUGH NORRIS
What is the name of your organizational unit?
  [Unknown]:  TRAINING
What is the name of your organization?
  [Unknown]:  I2TCH LIMITED
What is the name of your City or Locality?
  [Unknown]:  ADDLESTON
What is the name of your State or Province?
  [Unknown]:  SURREY
What is the two-letter country code for this unit?
  [Unknown]:  GB
Is CN=HUGH NORRIS, OU=TRAINING, O=I2TCH LIMITED, L=ADDLESTON, ST=SURREY, C=GB correct?
  [no]:  YES

Enter key password for <tomcat>
    (RETURN if same as keystore password): [Return]

Warning:
```

```
The JKS keystore uses a proprietary format. It is recommended to migrate to PKCS12 which is an industry standard
format using "keytool -importkeystore -srckeystore /usr/tomcat8/security/www_i2tch_loc.keystore -destkeystore
/usr/tomcat8/security/www_i2tch_loc.keystore -deststoretype pkcs12".
```

Après la création de la clef, il est nécessaire de créer un **CSR** (*Certificate Signing Request*). Pour créer le CSR, il convient d'utiliser de nouveau la commande **keytool** :

```
[root@centos7 bin]# keytool -certreq -keyalg RSA -alias tomcat -file www_i2tch_loc.csr -keystore
$CATALINA_HOME/security/www_i2tch_loc.keystore
Enter keystore password:  fenestros

Warning:
The JKS keystore uses a proprietary format. It is recommended to migrate to PKCS12 which is an industry standard
format using "keytool -importkeystore -srckeystore /usr/tomcat8/security/www_i2tch_loc.keystore -destkeystore
/usr/tomcat8/security/www_i2tch_loc.keystore -deststoretype pkcs12".
```

A ce stade, vous enverriez votre CSR à un organisme PKI tel **VeriSign** qui, après vérification des informations contenues dans votre CRT, signerait votre demande avec leur clef produisant ainsi un certificat qu'il vous retourne accompagné de son **Certificat Racine**.

Après réception de ce fichier, vous devez importer le **Certificat Racine** de votre PKI dans votre keystore, **par exemple** :

```
# keytool -import -alias root -keystore $CATALINA_HOME/security/www_i2tch_loc.keystore -file
<nom_du_certificat_racine>
```

Ensuite, vous devez importer votre propre certificat, **par exemple** :

```
# keytool -import -alias tomcat -keystore $CATALINA_HOME/security/www_i2tch_loc.keystore -trustcacerts -file
<nom_de_votre_certificat>
```

Pour plus d'informations concernant la création d'un keystore au format **.jks**, consultez cette page.

Dans le cas de ce LAB, nous n'allons pas faire appelle à un PKI. Par conséquent, il convient de signé notre propre CRT avec notre clef privée. Cette action génère un certificat SSL directement dans le keystore :

```
[root@centos7 bin]# keytool -selfcert -alias tomcat -keypass fenestros -keystore
$CATALINA_HOME/security/www_i2tch_loc.keystore -dname "CN=HUGH NORRIS, OU=TRAINING, O=I2TCH LIMITED, L=ADDLESTON,
ST=SURREY, C=GB"
Enter keystore password:  fenestros

Warning:
The JKS keystore uses a proprietary format. It is recommended to migrate to PKCS12 which is an industry standard
format using "keytool -importkeystore -srckeystore /usr/tomcat8/security/www_i2tch_loc.keystore -destkeystore
/usr/tomcat8/security/www_i2tch_loc.keystore -deststoretype pkcs12".
```

Dernièrement, ajoutez le connector suivant au fichier **$CATALINA_HOME/conf/server.xml** :

```
...

    <Connector executor="tomcatThreadPool"
               port="8080" protocol="HTTP/1.1"
               connectionTimeout="20000"
               redirectPort="8443" />

    <!-- Define a SSL/TLS HTTP/1.1 Connector on port 8443
         This connector uses the NIO implementation that requires the JSSE
         style configuration. When using the APR/native implementation, the
         OpenSSL style configuration is required as described in the APR/native
         documentation -->
    <!--
    <Connector port="8443" protocol="org.apache.coyote.http11.Http11NioProtocol"
               maxThreads="150" SSLEnabled="true" scheme="https" secure="true"
               clientAuth="false" sslProtocol="TLS" />
    -->

    <Connector port="8443" maxThreads="150" minSpareThreads="25" maxSpareThreads="75"
               enableLookups="false" disableUploadTimeout="true"
               acceptCount="100" debug="0" SSLEnabled="true" scheme="https" secure="true"
               clientAuth="false" sslProtocol="TLS"
```

```
               keystoreFile="/usr/tomcat8/security/www_i2tch_loc.keystore"
               keystorePass="fenestros" />
...
```

Redémarrez le serveur Tomcat :

```
[root@centos7 bin]# ./shutdown.sh
Using CATALINA_BASE:   /usr/tomcat8
Using CATALINA_HOME:   /usr/tomcat8
Using CATALINA_TMPDIR: /usr/tomcat8/temp
Using JRE_HOME:        /usr/lib/jvm/jre-1.8.0-openjdk-1.8.0.232.b09-0.el7_7.x86_64
Using CLASSPATH:       /usr/tomcat8/bin/bootstrap.jar:/usr/tomcat8/bin/tomcat-juli.jar
[root@centos7 bin]# ./startup.sh
Using CATALINA_BASE:   /usr/tomcat8
Using CATALINA_HOME:   /usr/tomcat8
Using CATALINA_TMPDIR: /usr/tomcat8/temp
Using JRE_HOME:        /usr/lib/jvm/jre-1.8.0-openjdk-1.8.0.232.b09-0.el7_7.x86_64
Using CLASSPATH:       /usr/tomcat8/bin/bootstrap.jar:/usr/tomcat8/bin/tomcat-juli.jar
Tomcat started.
```

Vérifiez maintenant que vous pouvez vous connecter avec le compte **admin3** sur le port **8443** en utilisant le navigateur Firefox et le lien **https://www.i2tch.loc:8443/manager/text/serverinfo**.

Configurer Apache

Installation de SSL

Installez le module **mod_ssl** pour Apache :

```
[root@centos7 bin]# rpm -qa | grep ssl
python-backports-ssl_match_hostname-3.5.0.1-1.el7.noarch
openssl-1.0.2k-19.el7.x86_64
```

```
openssl-libs-1.0.2k-19.el7.x86_64
xmlsec1-openssl-1.2.20-7.el7_4.x86_64
[root@centos7 bin]#
[root@centos7 bin]# yum install mod_ssl
...
[root@centos7 bin]# rpm -qa | grep ssl
python-backports-ssl_match_hostname-3.5.0.1-1.el7.noarch
openssl-1.0.2k-19.el7.x86_64
openssl-libs-1.0.2k-19.el7.x86_64
xmlsec1-openssl-1.2.20-7.el7_4.x86_64
mod_ssl-2.4.6-90.el7.centos.x86_64
```

Configuration de SSL

Dans le cas où vous souhaitez générer vos propres clés, vous devez d'abord générer une clé privée, nécessaire pour la création d'un **Certificate Signing Request**. Le CSR doit alors être envoyé à une des sociétés faisant autorité en la matière afin que celle-ci puisse vous retourner votre certificat définitif. Ce service est payant. C'est ce certificat définitif qui est utilisé pour des connexions sécurisées.

Saisissez donc la commande suivante pour générer votre clé privée :

```
[root@centos7 bin]# cd /root ; openssl genrsa -out www.i2tch.loc.key 1024
Generating RSA private key, 1024 bit long modulus
.............++++++
.......................................++++++
e is 65537 (0x10001)
```

Générer maintenant votre CSR :

```
[root@centos7 ~]# openssl req -new -key www.i2tch.loc.key -out www.i2tch.loc.csr
You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a DN.
There are quite a few fields but you can leave some blank
```

```
For some fields there will be a default value,
If you enter '.', the field will be left blank.
-----
Country Name (2 letter code) [XX]:GB
State or Province Name (full name) []:SURREY
Locality Name (eg, city) [Default City]:ADDLESTON
Organization Name (eg, company) [Default Company Ltd]:I2TCH LIMITED
Organizational Unit Name (eg, section) []:TRAINING
Common Name (eg, your name or your server's hostname) []:www.i2tch.loc
Email Address []:infos@i2tch.co.uk

Please enter the following 'extra' attributes
to be sent with your certificate request
A challenge password []: [RETURN]
An optional company name []: [RETURN]
```

et répondez aux questions qui vous sont posées. Notez bien la réponse à la question **Common Name**. Si vous ne donnez pas le nom de votre site, certains navigateurs ne géreront pas votre certificat correctement. Vous pouvez maintenant envoyé votre CSR à la société que vous avez choisie. Quand votre clé **.crt** vous est retournée, copiez-la, ainsi que votre clé privée dans le répertoire **/etc/pki/tls/certs/**.

Sans passer par un prestataire externe, vous pouvez signer votre CSR avec votre propre clé afin de générer votre certificat :

```
[root@centos7 ~]# openssl x509 -req -days 365 -in www.i2tch.loc.csr -signkey www.i2tch.loc.key -out
www.i2tch.loc.crt
Signature ok
subject=/C=GB/ST=SURREY/L=ADDLESTON/O=I2TCH LIMITED/OU=TRAINING/CN=www.i2tch.loc/emailAddress=infos@i2tch.eu
Getting Private key
```

Il convient ensuite de copier le certificat dans le répertoire **/etc/pki/tls/certs/** et la clef privée dans le répertoire **/etc/pki/tls/private/** :

```
[root@centos7 ~]# cp /root/www.i2tch.loc.key /etc/pki/tls/private/
[root@centos7 ~]# cp /root/www.i2tch.loc.crt /etc/pki/tls/certs/
```

Mise en place des paramètres de sécurité SSL

Consultez le contenu du répertoire **/etc/httpd/conf.d/** :

```
[root@centos7 ~]# ls /etc/httpd/conf.d
autoindex.conf  README  ssl.conf  userdir.conf  welcome.conf
```

Ce répertoire contient des fichiers dont le contenu est inclut dans le corps du fichier httpd.conf.

Sauvegardez le fichier **ssl.conf**.

```
[root@centos7 ~]# cp /etc/httpd/conf.d/ssl.conf /root/ssl.conf.backup
```

Ouvrez le fichier /etc/httpd/conf.d/ssl.conf et modifiez la ligne suivante :

```
#DocumentRoot "/var/www/html"
```

en :

```
DocumentRoot "/var/www/html"
```

Cette directive indique que la racine du site sécurisé sera **/var/www/html**.

Deuxièmement, ajoutez la ligne suivante en dessous de la directive **#ServerName …** existante :

```
ServerName www.i2tch.loc:443
```

Dernièrement modifiez les deux lignes suivantes :

```
SSLCertificateFile /etc/pki/tls/certs/localhost.crt
```

```
SSLCertificateKeyFile /etc/pki/tls/private/localhost.key
```

en :

```
SSLCertificateFile /etc/pki/tls/certs/www.i2tch.loc.crt
```

```
SSLCertificateKeyFile /etc/pki/tls/private/www.i2tch.loc.key
```

Vous obtiendrez :

```
...
#   Server Certificate:
# Point SSLCertificateFile at a PEM encoded certificate.  If
# the certificate is encrypted, then you will be prompted for a
# pass phrase.  Note that a kill -HUP will prompt again.  A new
# certificate can be generated using the genkey(1) command.
#SSLCertificateFile /etc/pki/tls/certs/localhost.crt
SSLCertificateFile /etc/pki/tls/certs/www.i2tch.loc.crt

#   Server Private Key:
#   If the key is not combined with the certificate, use this
#   directive to point at the key file.  Keep in mind that if
#   you've both a RSA and a DSA private key you can configure
#   both in parallel (to also allow the use of DSA ciphers, etc.)
# SSLCertificateKeyFile /etc/pki/tls/private/localhost.key
SSLCertificateKeyFile /etc/pki/tls/private/www.i2tch.loc.key
...
```

Sauvegardez le fichier et redémarrez le serveur apache :

```
[root@centos7 ~]# systemctl restart httpd.service
```

Tester Votre Configuration

Pour tester votre serveur apache en mode SSL, vous allez procéder à deux tests distincts.

Dans le premier, saisissez la commande suivante en ligne de commande et en tant que root :

```
[root@centos7 ~]# openssl s_client -connect www.i2tch.loc:443
CONNECTED(00000003)
depth=0 C = GB, ST = SURREY, L = ADDLESTON, O = I2TCH LIMITED, OU = TRAINING, CN = www.i2tch.loc, emailAddress =
infos@i2tch.eu
verify error:num=18:self signed certificate
verify return:1
depth=0 C = GB, ST = SURREY, L = ADDLESTON, O = I2TCH LIMITED, OU = TRAINING, CN = www.i2tch.loc, emailAddress =
infos@i2tch.eu
verify return:1
---
Certificate chain
 0 s:/C=GB/ST=SURREY/L=ADDLESTON/O=I2TCH LIMITED/OU=TRAINING/CN=www.i2tch.loc/emailAddress=infos@i2tch.eu
   i:/C=GB/ST=SURREY/L=ADDLESTON/O=I2TCH LIMITED/OU=TRAINING/CN=www.i2tch.loc/emailAddress=infos@i2tch.eu
---
Server certificate
-----BEGIN CERTIFICATE-----
MIICoTCCAgoCCQD2lg50Gvmh/jANBgkqhkiG9w0BAQUFADCBlDELMAkGA1UEBhMC
R0IxDzANBgNVBAgMBlNVUlJFWTESMBAGA1UEBwwJQURETEVTVE9OMRYwFAYDVQQK
DA1JMlRDSCBMSU1JVEVEMREwDwYDVQQLDAhUUkFJTklORzEWMBQGA1UEAwwNd3d3
LmkydGNoLmxvYzEdMBsGCSqGSIb3DQEJARYOaW5mb3NAaTJ0Y2guZXUwHhcNMTYw
NjI4MTYyNDMxWhcNMTcwNjI4MTYyNDMxWjCBlDELMAkGA1UEBhMCR0IxDzANBgNV
BAgMBlNVUlJFWTESMBAGA1UEBwwJQURETEVTVE9OMRYwFAYDVQQKDA1JMlRDSCBM
SU1JVEVEMREwDwYDVQQLDAhUUkFJTklORzEWMBQGA1UEAwwNd3d3LmkydGNoLmxv
YzEdMBsGCSqGSIb3DQEJARYOaW5mb3NAaTJ0Y2guZXUwgZ8wDQYJKoZIhvcNAQEB
BQADgY0AMIGJAoGBAKRfbHwr4FNll+cwYN7gkIIp30EhwOgZU6ql/4K49zw54Sk9
vZ2uhECcrtHcdVwGJtieLjDVWoCI4RIflv6XXvHdUbScpRKHadrQDtvySHAzLw9L
SDbOGj26zK8dHFUCE2lDVyVGVWHgvgdCtrYJxDjwqZlmCOqA9Ii5587h8McpAgMB
AAEwDQYJKoZIhvcNAQEFBQADgYEAJiGMZlGh8a+DfzXGI9fwoX+kS/nSNotS5D0j
NaVM97NCwAtH/DX4qUYUp1UDmIpCEaZcO8e9Xt/GDjkJRSb6SqufFREbs1kRfKzW
3ar2Xnnt0Qr73YXr03uMP+/WdmIDF9NawLZ7C37m+KgnUJQlLuE+nUMjeQI1ogYM
```

```
KdTjblI=
-----END CERTIFICATE-----
subject=/C=GB/ST=SURREY/L=ADDLESTON/O=I2TCH LIMITED/OU=TRAINING/CN=www.i2tch.loc/emailAddress=infos@i2tch.eu
issuer=/C=GB/ST=SURREY/L=ADDLESTON/O=I2TCH LIMITED/OU=TRAINING/CN=www.i2tch.loc/emailAddress=infos@i2tch.eu
---
No client certificate CA names sent
Server Temp Key: ECDH, secp521r1, 521 bits
---
SSL handshake has read 1308 bytes and written 441 bytes
---
New, TLSv1/SSLv3, Cipher is ECDHE-RSA-AES256-GCM-SHA384
Server public key is 1024 bit
Secure Renegotiation IS supported
Compression: NONE
Expansion: NONE
SSL-Session:
    Protocol  : TLSv1.2
    Cipher    : ECDHE-RSA-AES256-GCM-SHA384
    Session-ID: C8B3E6709FCB21D5014F0938AA73F32E1D3F73A6B38FF3A238032CD268808916
    Session-ID-ctx:
    Master-Key: 798987A0DADE2C89E7915B789A14979A3FDC4D9371A37A916273CE98890DB9978B401072867053FC9178B3453F219360
    Key-Arg   : None
    Krb5 Principal: None
    PSK identity: None
    PSK identity hint: None
    TLS session ticket lifetime hint: 300 (seconds)
    TLS session ticket:
    0000 - 40 e7 3a 85 fe 5f e5 a2-60 cd 51 4e 1b 49 f9 d5   @.:.._..`.QN.I..
    0010 - b3 65 8c 63 72 b3 60 1c-47 ed 83 5e ae 42 55 da   .e.cr.`.G..^.BU.
    0020 - f6 7f 30 1d 3b 97 a8 09-31 25 35 58 24 f4 ee a3   ..0.;...1%5X$...
    0030 - d7 a1 19 30 be e5 6e 93-4a ec 19 ae c0 49 85 5b   ...0..n.J....I.[
    0040 - 5c 0f bf 04 f8 31 b1 92-99 0c f3 cf fd 85 42 83   \....1........B.
    0050 - 07 2f 31 21 c9 f1 44 19-57 1a 6c c4 20 dc 5f 22   ./1!..D.W.l. ._"
    0060 - 11 6d ac cf 60 fc 68 20-4e ac 03 f7 55 f6 31 25   .m..`.h N...U.1%
```

```
    0070 - 64 91 dc fc be 02 47 0d-46 a5 ad cd 7b a4 69 41   d.....G.F...{.iA
    0080 - 5c 32 25 cf fe 1c af cb-fc 7a 0d 33 47 e0 43 93   \2%......z.3G.C.
    0090 - b3 57 73 e1 69 30 cd 25-1e 0d b8 74 13 66 93 01   .Ws.i0.%...t.f..
    00a0 - aa 29 3d 25 32 59 4e ad-3d 00 ca 26 48 7b f0 bc   .)=%2YN.=..&H{..
    00b0 - 30 8c ed d9 e9 ae 23 26-f1 a4 ee 35 91 16 82 ff   0.....#&...5....

    Start Time: 1467131982
    Timeout   : 300 (sec)
    Verify return code: 18 (self signed certificate)
---
^C
```

Notez qu'il y a génération d'erreurs. Ceci est normal. Ce test démontre que votre site sécurisé fonctionne. Votre serveur apache a été configuré avec succès.

Apache en Frontal HTTPS

Pour utiliser le serveur web Apache en tant que serveur frontal HTTPS pour Tomcat, il convient d'utiliser le proxy d'apache. Vérifiez donc que les deux lignes **LoadModule proxy_module modules/mod_proxy.so** et **LoadModule proxy_http_module modules/mod_proxy_http.so** soient bien présentes dans le fichier **/etc/httpd/conf.modules.d/00-proxy.conf** :

```
[root@centos7 ~]# cat /etc/httpd/conf.modules.d/00-proxy.conf
# This file configures all the proxy modules:
LoadModule proxy_module modules/mod_proxy.so
LoadModule lbmethod_bybusyness_module modules/mod_lbmethod_bybusyness.so
LoadModule lbmethod_byrequests_module modules/mod_lbmethod_byrequests.so
LoadModule lbmethod_bytraffic_module modules/mod_lbmethod_bytraffic.so
LoadModule lbmethod_heartbeat_module modules/mod_lbmethod_heartbeat.so
LoadModule proxy_ajp_module modules/mod_proxy_ajp.so
LoadModule proxy_balancer_module modules/mod_proxy_balancer.so
LoadModule proxy_connect_module modules/mod_proxy_connect.so
LoadModule proxy_express_module modules/mod_proxy_express.so
LoadModule proxy_fcgi_module modules/mod_proxy_fcgi.so
```

```
LoadModule proxy_fdpass_module modules/mod_proxy_fdpass.so
LoadModule proxy_ftp_module modules/mod_proxy_ftp.so
LoadModule proxy_http_module modules/mod_proxy_http.so
LoadModule proxy_scgi_module modules/mod_proxy_scgi.so
LoadModule proxy_wstunnel_module modules/mod_proxy_wstunnel.so
```

Ajoutez maintenant les directives suivantes à la fin du fichier **/etc/httpd/conf.d/ssl.conf** juste avant la balise </VirtualHost> :

```
[root@centos7 ~]# cat /etc/httpd/conf.d/ssl.conf
...
#   Per-Server Logging:
#   The home of a custom SSL log file. Use this when you want a
#   compact non-error SSL logfile on a virtual host basis.
CustomLog logs/ssl_request_log \
          "%t %h %{SSL_PROTOCOL}x %{SSL_CIPHER}x \"%r\" %b"
<IfModule mod_proxy.c>
    ProxyRequests                     Off
    ProxyPreserveHost                 On
    ProxyPass           /docs         http://www.i2tch.loc:8443/docs
    ProxyPassReverse    /docs         http://www.i2tch.loc:8443/docs
</IfModule>

</VirtualHost>
```

Sauvegardez et relancez le service httpd :

```
[root@centos7 ~]# systemctl restart httpd.service
```

Éditez maintenant le Connector HTTPS du fichier **$CATALINA_HOME/conf/server.xml** :

```
...
    <!--
    <Connector port="8443" maxThreads="150" minSpareThreads="25" maxSpareThreads="75"
               enableLookups="false" disableUploadTimeout="true"
```

```
               acceptCount="100" debug="0" scheme="https" secure="true"
               clientAuth="false" sslProtocol="TLS"
               keystoreFile="/usr/tomcat8/security/www_i2tch_loc.keystore"
               keystorePass="fenestros" />
    -->

    <Connector port="8443" proxyPort="443" proxyName="10.0.3.51" />
...
```

Sauvegardez et relancez le service tomcat :

```
[root@centos7 ~]# cd $CATALINA_HOME/bin
[root@centos7 bin]# ./shutdown.sh
Using CATALINA_BASE:   /usr/tomcat8
Using CATALINA_HOME:   /usr/tomcat8
Using CATALINA_TMPDIR: /usr/tomcat8/temp
Using JRE_HOME:        /usr/lib/jvm/jre-1.8.0-openjdk-1.8.0.232.b09-0.el7_7.x86_64
Using CLASSPATH:       /usr/tomcat8/bin/bootstrap.jar:/usr/tomcat8/bin/tomcat-juli.jar
[root@centos7 bin]# ./startup.sh
Using CATALINA_BASE:   /usr/tomcat8
Using CATALINA_HOME:   /usr/tomcat8
Using CATALINA_TMPDIR: /usr/tomcat8/temp
Using JRE_HOME:        /usr/lib/jvm/jre-1.8.0-openjdk-1.8.0.232.b09-0.el7_7.x86_64
Using CLASSPATH:       /usr/tomcat8/bin/bootstrap.jar:/usr/tomcat8/bin/tomcat-juli.jar
Tomcat started.
```

A ce stade, le serveur Tomcat ne propose plus de <Connector> direct en https. Par contre, grâce à la configuration du proxy apache, les connexions à l'application **docs** seront cryptées par le SSL d'apache.

Afin de vous assurer que la configuration est bien faite, saisissez l'URL suivant dans le navigateur de votre VM : **https://www.i2tch.loc/docs**.

Restrictions d'Accès

Les restrictions d'accès de Tomcat permet la mise en place de restrictions pour :

- le Serveur,
- un hôte particulier de ce serveur,
- une application.

Les Valves utilisent les classes **org.apache.catalina.valves.RemoteHostValve** et **org.apache.catalina.valves.RemoteAddrValve**.

Les deux filtres ci-dessus utilisent les même attributs :

- **allow**,
- **deny**.

Les deux attributs prennent en tant que valeur une expression régulière au format **java.util.regex** identifiant des adresses IP ou des noms d'hôtes.

Le dernier attribut est **denyStatus** qui permet de spécifier quel code d'erreur HTML sera envoyé vers le navigateur du client. Par défaut cette valeur est **403** *Forbidden*.

Voici deux exemples de restrictions :

```
    <Valve className='org.apache.catelina.valves.RemoteAddrValve"
        allow="127\.0\.0\.1|10\.\d+\.\d+" />
    </Valve>
```

```
    <Valve className='org.apache.catelina.valves.RemoteHostValve"
        allow="\w+\.i2tch\.loc" />
    </Valve>
```

Le Gestionnaire de Sécurité

La machine virtuelle Java dispose d'une classe Java spéciale appelée **SecurityManager** (*Gestionnaire de Sécurité*). Cette classe permet de bloquer

l'exécution de certaines classes ainsi que de bloquer l'accès à des ressources système aux classes qui s'exécutent. Une fois activé le Gestionnaire de Sécurité interdit tout en utilisant des fichiers de configuration qui se terminent en général par l'extension **.policy**.

Une directive d'un fichier .policy prend une syntaxe particulière :

```
grant [codeBase <code>] {
    permission <classe> [<nom>, <liste permissions>];
};
```

Par exemple pour autoriser **uniquement** la lecture du fichier **/tmp/fichier** par tout le code du répertoire **$JAVA_HOME/lib/ext**, la directive devient :

```
grant codeBase "file:${java.home}/lib/ext/*" {
    permission java.io.FilePermission "/tmp/fichier", "read";
};
```

Dans cette directive, la portée du codeBase diffère selon l'écriture de la clause **file:** :

- **file:${java.home}/lib/ext/**,
  - concerne uniquement les classes dans ${java.home}/lib/ext/ mais pas les classes et fichiers JAR des sous-répertoires,
- **file:${java.home}/lib/ext/*** ,
  - concerne les classes et le fichiers JAR dans ${java.home}/lib/ext/ mais pas les classes et fichiers JAR des sous-répertoires,
- **file:${java.home}/lib/ext/-**,
  - * concerne les classes et le fichiers JAR dans ${java.home}/lib/ext/ et celles et ceux dans des sous-répertoires.

> **Important** : Il est aussi possible d'utiliser **http:** à la place de **file:**.

La liste des permissions définies en standard est :

- **java.security.AllPermission**,
- **java.security.SecurityPermission**,
- **java.io.SerializablePermission**,
- **java.lang.reflect.ReflectPermission**,

- **java.lang.RuntimePermission**,
- **java.net.NetPermission**,
- **java.net.SocketPermission**,
- **java.util.PropertyPermission**.

Le fichier .policy chargé par défaut lors de l'activation du Gestionnaire de Sécurité est **$JAVA_HOME/lib/security/java.policy** :

```
[root@centos7 bin]# cat $JAVA_HOME/lib/security/java.policy

// Standard extensions get all permissions by default

grant codeBase "file:${{java.ext.dirs}}/*" {
        permission java.security.AllPermission;
};

// default permissions granted to all domains

grant {
        // Allows any thread to stop itself using the java.lang.Thread.stop()
        // method that takes no argument.
        // Note that this permission is granted by default only to remain
        // backwards compatible.
        // It is strongly recommended that you either remove this permission
        // from this policy file or further restrict it to code sources
        // that you specify, because Thread.stop() is potentially unsafe.
        // See the API specification of java.lang.Thread.stop() for more
        // information.
        permission java.lang.RuntimePermission "stopThread";

        // allows anyone to listen on dynamic ports
        permission java.net.SocketPermission "localhost:0", "listen";

        // "standard" properies that can be read by anyone
```

```
        permission java.util.PropertyPermission "java.version", "read";
        permission java.util.PropertyPermission "java.vendor", "read";
        permission java.util.PropertyPermission "java.vendor.url", "read";
        permission java.util.PropertyPermission "java.class.version", "read";
        permission java.util.PropertyPermission "os.name", "read";
        permission java.util.PropertyPermission "os.version", "read";
        permission java.util.PropertyPermission "os.arch", "read";
        permission java.util.PropertyPermission "file.separator", "read";
        permission java.util.PropertyPermission "path.separator", "read";
        permission java.util.PropertyPermission "line.separator", "read";

        permission java.util.PropertyPermission "java.specification.version", "read";
        permission java.util.PropertyPermission "java.specification.vendor", "read";
        permission java.util.PropertyPermission "java.specification.name", "read";

        permission java.util.PropertyPermission "java.vm.specification.version", "read";
        permission java.util.PropertyPermission "java.vm.specification.vendor", "read";
        permission java.util.PropertyPermission "java.vm.specification.name", "read";
        permission java.util.PropertyPermission "java.vm.version", "read";
        permission java.util.PropertyPermission "java.vm.vendor", "read";
        permission java.util.PropertyPermission "java.vm.name", "read";
};
```

Pour activer le Gestionnaire de Sécurité, il faut démarrer la machine virtuelle Java avec l'option **-Djava.security.manager**. Ceci est déjà prévu sous Tomcat. En effet il suffit de passer l'option **-security** au script **$CATALINA_HOME/bin/startup.sh**. Dans ce cas c'est le contenu du fichier **$CATALINA_HOME/conf/catalina.policy** qui est appliqué :

```
[root@centos7 bin]# cat $CATALINA_HOME/conf/catalina.policy
// Licensed to the Apache Software Foundation (ASF) under one or more
// contributor license agreements.  See the NOTICE file distributed with
// this work for additional information regarding copyright ownership.
// The ASF licenses this file to You under the Apache License, Version 2.0
// (the "License"); you may not use this file except in compliance with
// the License.  You may obtain a copy of the License at
```

```
//
//     http://www.apache.org/licenses/LICENSE-2.0
//
// Unless required by applicable law or agreed to in writing, software
// distributed under the License is distributed on an "AS IS" BASIS,
// WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
// See the License for the specific language governing permissions and
// limitations under the License.

// ============================================================================
// catalina.policy - Security Policy Permissions for Tomcat
//
// This file contains a default set of security policies to be enforced (by the
// JVM) when Catalina is executed with the "-security" option.  In addition
// to the permissions granted here, the following additional permissions are
// granted to each web application:
//
// * Read access to the web application's document root directory
// * Read, write and delete access to the web application's working directory
// ============================================================================


// ========== SYSTEM CODE PERMISSIONS =========================================


// These permissions apply to javac
grant codeBase "file:${java.home}/lib/-" {
        permission java.security.AllPermission;
};

// These permissions apply to all shared system extensions
grant codeBase "file:${java.home}/jre/lib/ext/-" {
        permission java.security.AllPermission;
};
```

```
// These permissions apply to javac when ${java.home] points at $JAVA_HOME/jre
grant codeBase "file:${java.home}/../lib/-" {
        permission java.security.AllPermission;
};

// These permissions apply to all shared system extensions when
// ${java.home} points at $JAVA_HOME/jre
grant codeBase "file:${java.home}/lib/ext/-" {
        permission java.security.AllPermission;
};


// ========== CATALINA CODE PERMISSIONS =======================================


// These permissions apply to the daemon code
grant codeBase "file:${catalina.home}/bin/commons-daemon.jar" {
        permission java.security.AllPermission;
};

// These permissions apply to the logging API
// Note: If tomcat-juli.jar is in ${catalina.base} and not in ${catalina.home},
// update this section accordingly.
//  grant codeBase "file:${catalina.base}/bin/tomcat-juli.jar" {..}
grant codeBase "file:${catalina.home}/bin/tomcat-juli.jar" {
        permission java.io.FilePermission
         "${java.home}${file.separator}lib${file.separator}logging.properties", "read";

        permission java.io.FilePermission
         "${catalina.base}${file.separator}conf${file.separator}logging.properties", "read";
        permission java.io.FilePermission
         "${catalina.base}${file.separator}logs", "read, write";
        permission java.io.FilePermission
         "${catalina.base}${file.separator}logs${file.separator}*", "read, write";
```

```
        permission java.lang.RuntimePermission "shutdownHooks";
        permission java.lang.RuntimePermission "getClassLoader";
        permission java.lang.RuntimePermission "setContextClassLoader";

        permission java.lang.management.ManagementPermission "monitor";

        permission java.util.logging.LoggingPermission "control";

        permission java.util.PropertyPermission "java.util.logging.config.class", "read";
        permission java.util.PropertyPermission "java.util.logging.config.file", "read";
        permission java.util.PropertyPermission "org.apache.juli.AsyncLoggerPollInterval", "read";
        permission java.util.PropertyPermission "org.apache.juli.AsyncMaxRecordCount", "read";
        permission java.util.PropertyPermission "org.apache.juli.AsyncOverflowDropType", "read";
        permission java.util.PropertyPermission "org.apache.juli.ClassLoaderLogManager.debug", "read";
        permission java.util.PropertyPermission "catalina.base", "read";

        // Note: To enable per context logging configuration, permit read access to
        // the appropriate file. Be sure that the logging configuration is
        // secure before enabling such access.
        // E.g. for the examples web application (uncomment and unwrap
        // the following to be on a single line):
        // permission java.io.FilePermission "${catalina.base}${file.separator}
        //  webapps${file.separator}examples${file.separator}WEB-INF
        //  ${file.separator}classes${file.separator}logging.properties", "read";
};

// These permissions apply to the server startup code
grant codeBase "file:${catalina.home}/bin/bootstrap.jar" {
        permission java.security.AllPermission;
};

// These permissions apply to the servlet API classes
// and those that are shared across all class loaders
// located in the "lib" directory
```

```
grant codeBase "file:${catalina.home}/lib/-" {
        permission java.security.AllPermission;
};


// If using a per instance lib directory, i.e. ${catalina.base}/lib,
// then the following permission will need to be uncommented
// grant codeBase "file:${catalina.base}/lib/-" {
//         permission java.security.AllPermission;
// };


// ========== WEB APPLICATION PERMISSIONS =====================================


// These permissions are granted by default to all web applications
// In addition, a web application will be given a read FilePermission
// for all files and directories in its document root.
grant {
    // Required for JNDI lookup of named JDBC DataSource's and
    // javamail named MimePart DataSource used to send mail
    permission java.util.PropertyPermission "java.home", "read";
    permission java.util.PropertyPermission "java.naming.*", "read";
    permission java.util.PropertyPermission "javax.sql.*", "read";

    // OS Specific properties to allow read access
    permission java.util.PropertyPermission "os.name", "read";
    permission java.util.PropertyPermission "os.version", "read";
    permission java.util.PropertyPermission "os.arch", "read";
    permission java.util.PropertyPermission "file.separator", "read";
    permission java.util.PropertyPermission "path.separator", "read";
    permission java.util.PropertyPermission "line.separator", "read";

    // JVM properties to allow read access
```

```
    permission java.util.PropertyPermission "java.version", "read";
    permission java.util.PropertyPermission "java.vendor", "read";
    permission java.util.PropertyPermission "java.vendor.url", "read";
    permission java.util.PropertyPermission "java.class.version", "read";
    permission java.util.PropertyPermission "java.specification.version", "read";
    permission java.util.PropertyPermission "java.specification.vendor", "read";
    permission java.util.PropertyPermission "java.specification.name", "read";

    permission java.util.PropertyPermission "java.vm.specification.version", "read";
    permission java.util.PropertyPermission "java.vm.specification.vendor", "read";
    permission java.util.PropertyPermission "java.vm.specification.name", "read";
    permission java.util.PropertyPermission "java.vm.version", "read";
    permission java.util.PropertyPermission "java.vm.vendor", "read";
    permission java.util.PropertyPermission "java.vm.name", "read";

    // Required for OpenJMX
    permission java.lang.RuntimePermission "getAttribute";

    // Allow read of JAXP compliant XML parser debug
    permission java.util.PropertyPermission "jaxp.debug", "read";

    // All JSPs need to be able to read this package
    permission java.lang.RuntimePermission "accessClassInPackage.org.apache.tomcat";

    // Precompiled JSPs need access to these packages.
    permission java.lang.RuntimePermission "accessClassInPackage.org.apache.jasper.el";
    permission java.lang.RuntimePermission "accessClassInPackage.org.apache.jasper.runtime";
    permission java.lang.RuntimePermission
     "accessClassInPackage.org.apache.jasper.runtime.*";

    // Precompiled JSPs need access to these system properties.
    permission java.util.PropertyPermission
     "org.apache.jasper.runtime.BodyContentImpl.LIMIT_BUFFER", "read";
    permission java.util.PropertyPermission
```

```
     "org.apache.el.parser.COERCE_TO_ZERO", "read";

    // The cookie code needs these.
    permission java.util.PropertyPermission
     "org.apache.catalina.STRICT_SERVLET_COMPLIANCE", "read";
    permission java.util.PropertyPermission
     "org.apache.tomcat.util.http.ServerCookie.STRICT_NAMING", "read";
    permission java.util.PropertyPermission
     "org.apache.tomcat.util.http.ServerCookie.FWD_SLASH_IS_SEPARATOR", "read";

    // Applications using Comet need to be able to access this package
    permission java.lang.RuntimePermission "accessClassInPackage.org.apache.catalina.comet";

    // Applications using WebSocket need to be able to access these packages
    permission java.lang.RuntimePermission "accessClassInPackage.org.apache.tomcat.websocket";
    permission java.lang.RuntimePermission "accessClassInPackage.org.apache.tomcat.websocket.server";
};


// The Manager application needs access to the following packages to support the
// session display functionality. These settings support the following
// configurations:
// - default CATALINA_HOME == CATALINA_BASE
// - CATALINA_HOME != CATALINA_BASE, per instance Manager in CATALINA_BASE
// - CATALINA_HOME != CATALINA_BASE, shared Manager in CATALINA_HOME
grant codeBase "file:${catalina.base}/webapps/manager/-" {
    permission java.lang.RuntimePermission "accessClassInPackage.org.apache.catalina";
    permission java.lang.RuntimePermission "accessClassInPackage.org.apache.catalina.ha.session";
    permission java.lang.RuntimePermission "accessClassInPackage.org.apache.catalina.manager";
    permission java.lang.RuntimePermission "accessClassInPackage.org.apache.catalina.manager.util";
    permission java.lang.RuntimePermission "accessClassInPackage.org.apache.catalina.util";
};
grant codeBase "file:${catalina.home}/webapps/manager/-" {
    permission java.lang.RuntimePermission "accessClassInPackage.org.apache.catalina";
```

```
    permission java.lang.RuntimePermission "accessClassInPackage.org.apache.catalina.ha.session";
    permission java.lang.RuntimePermission "accessClassInPackage.org.apache.catalina.manager";
    permission java.lang.RuntimePermission "accessClassInPackage.org.apache.catalina.manager.util";
    permission java.lang.RuntimePermission "accessClassInPackage.org.apache.catalina.util";
};

// You can assign additional permissions to particular web applications by
// adding additional "grant" entries here, based on the code base for that
// application, /WEB-INF/classes/, or /WEB-INF/lib/ jar files.
//
// Different permissions can be granted to JSP pages, classes loaded from
// the /WEB-INF/classes/ directory, all jar files in the /WEB-INF/lib/
// directory, or even to individual jar files in the /WEB-INF/lib/ directory.
//
// For instance, assume that the standard "examples" application
// included a JDBC driver that needed to establish a network connection to the
// corresponding database and used the scrape taglib to get the weather from
// the NOAA web server.  You might create a "grant" entries like this:
//
// The permissions granted to the context root directory apply to JSP pages.
// grant codeBase "file:${catalina.base}/webapps/examples/-" {
//      permission java.net.SocketPermission "dbhost.mycompany.com:5432", "connect";
//      permission java.net.SocketPermission "*.noaa.gov:80", "connect";
// };
//
// The permissions granted to the context WEB-INF/classes directory
// grant codeBase "file:${catalina.base}/webapps/examples/WEB-INF/classes/-" {
// };
//
// The permission granted to your JDBC driver
// grant codeBase "jar:file:${catalina.base}/webapps/examples/WEB-INF/lib/driver.jar!/-" {
//      permission java.net.SocketPermission "dbhost.mycompany.com:5432", "connect";
// };
// The permission granted to the scrape taglib
```

```
// grant codeBase "jar:file:${catalina.base}/webapps/examples/WEB-INF/lib/scrape.jar!/-" {
//      permission java.net.SocketPermission "*.noaa.gov:80", "connect";
// };
```
