JNDI startTLs HowTo

Note: Nowadays StartTLS support is implemented in JDNIRealm of Tomcat — starting with Tomcat 7.0.60, 8.0.21 (BZ 49785).

This old page describes an alternative solution and is kept as a historic reference. Note that BZ 49785 has a link to this page.

JNDI StartTLS HowTo

In reference to: http://www.mail-archive.com/users@tomcat.apache.org/msg80660.html this Howto describes the configuration of a JNDI Realm connecting to an LDAP directory using StartTLS for connection establishment.

StartTLS is the method of negotiating a TLS connection. For LDAP it was first time in RFC 2830, then refined in RFC 4513.

Tomcat does not support this out of the box. Using JNDI Realm's `contextFactory` feature however, we can still achieve this:

```
<Realm className="org.apache.catalina.realm.JNDIRealm"
     connectionURL="ldap://primary.ldap.dir:389"
     alternateURL="ldap://secondary.ldap.dir:389"
     connectionName="uid=binddn" connectionPassword="password."
     userBase="ou=people,dc=brainsware,dc=org" userSearch="uid={0}"
     contextFactory="tc.startTLS.LdapTlsContextFactory />
```

Using the code provided by Felix Schumacher in this post: http://www.mail-archive.com/users@tomcat.apache.org/msg80693.html - You can download it here: LdapTlsContextFactory.java. We have to compile it into a JAR and put in a place where Tomcat can find it: `lib`. Then we simply reference its full name in `contextFactory`. `LdapTlsContextFactory` will now do the negotiation initialization. Afterwards the created object will be used for every authentication attempt. Beware that the code will not check the hostname of the server with respect to its certificate. If you don't want this behaviour remove the call to `tls.setHostNameVerifier(...)`.

Further Steps

The code probably needs auditing. More testing. And definitely more tightening: e.g.: When starting the negotiation the client (Tomcat + `LdapTlsContextFactory`) sends an `SSLv2Hello`, which is anything but desirable. This could be due to Sun’s poor defaults in their SSL implementation, an oversight in the code, or because I’ve missed out a JVM startup options.