LDAP for the Java™ Net URL Framework

Part I: The IAIK-JCE LdapURLConnection

Dieter.Bratko@iaik.tugraz.at
Stiftung Secure Information and Communication Technologies
IAIK, Graz University of Technology

Introduction

One of the basic requirements of an X.509 Public Key Infrastructure (PKI) is the public availability of certificates and certificate revocation lists (CRLs) from online repositories like HTTP servers or LDAP directories.

It is quite easy to use the Java™ URL framework to download a certificate or crl from an HTTP location. However, the default JDK does not provide an LDAP equivalent to the `HttpURLConnection` class of the `java.net` package. The means of choice for connecting to and searching an LDAP directory is the Java™ Naming And Directory Interface (JNDI). JNDI is a powerful tool, but using it requires some certain knowledge about X.500 and LDAP directory models and the JNDI API itself.

In this article we introduce the `LdapURLConnection` implementation of the IAIK-JCE crypto toolkit. It hides the required JNDI details from the application programmer and allows you to search for and download certificates and crls from LDAP repositories in an easy and straightforward way. We first will a brief description of the LDAP URL format, then compare the usage of `HttpURLConnection` and `LdapURLConnection`, and finally provide some source samples for downloading certificates and crls from HTTP and LDAP locations, respectively.

The LDAP URL format

LDAP directories are organized in a hierarchical, tree-like manner. At the node entries the information is stored in form of attributes (type-and-value pairs). The path from the root to some particular node of the tree may represent the distinguished name of the holder of the certificate stored at this node.

An LDAP URL as defined in RFC 2255 starts with the “ldap://” scheme prefix followed by host name and (optional) port number of the LDAP server you want to connect to:

```
ldapurl = scheme "://" [hostport] ["/"
                  [dn ["?" [attributes] ["?" [scope]
                  ["?" [filter] ["?" extensions]]]]]]
```

The components behind the `[hostport]` part have the following meaning:

- `dn`: the base distinguished name from which you want to start the LDAP search (default: empty string). For instance, the base dn may represent the country (e.g. `"c=at"`) in which the owner of the certificate you are searching for is living.
- `attributes:` may be set to tell the LDAP server which kind of attributes shall be returned as result of the search (e.g. `"caCertificate;binary"` or `"userCertificate;binary"` to return attributes that contain DER encoded ca- or end user certificates, respectively; or `"certificateRevocationList;binary"` to get certificate revocation lists only).
- `scope`: defines the search scope; "base" for base object search (default), "one" for one-level search, or "sub" for sub-tree search.
- `filter`: may be specified to search for entries containing attributes that match some certain criteria – e.g.: "(mail=Joe.TestUser@iaik.tugraz.at)" to search for entries that have a mail attribute with value "Joe.TestUser@iaik.tugraz.at".
- `extensions:` may be used in the future to extend the capabilities of the url

An LDAP url may look like, e.g. `"ldap://ldap.signatur.rtr.at/CN=Telekom-Control-Kommission%20Top%201, O=Telekom-Control-Kommission, C=AT?caCertificate;binary"` (for doing a base object search on the LDAP server at `ldap.signatur.rtr.at` to retrieve the top ca certificate of the Austrian regulation authority) or, for instance, `"ldap://ldap.signatur.rtr.at/CN=Certificate%20Revocation%201, O=Telekom-Control-Kommission, C=AT?certificateRevocationList;binary"` (to download the referenced certificate revocation list from the RTR LDAP server).

When using JNDI you now would have to parse the LDAP URL and then deal with JNDI `DirContext`, `SearchControl` and `NamingOperation` objects to get the certificate or crl from the LDAP server. In the next chapter we present the IAIK-JCE LdapURLConnection that does all the required JNDI processing and let you get a certificate or crl in quite the same simple way as when downloading it from an http server.

HttpURLConnection versus LdapURLConnection

The `java.net` URL framework provides a very easy approach to download a resource from an http location. First you create a `URL` object for the http location you want to connect to. Then you call method `openConnection()` on the `URL` object to get an input stream for reading the certificate or crl from it:

```
// create HTTP URL
URL url = new URL("http://www.signatur.rtr.at/currenttop.cer");
// open connection
URLConnection con = url.openConnection();
// get a stream and read the certificate
InputStream is = con.getInputStream();
X509Certificate cert = new X509Certificate(is);
```

Now try the same with an LDAP URL:

```
// create LDAP URL
String urlStr = "ldap://ldap.signatur.rtr.at/CN=Telekom-Control-"+
                "Kommission%20Top%201,O=Telekom-Control-Kommission,"+
                "C=AT?caCertificate;binary";
URL url = new URL(urlStr);
// open connection
URLConnection con = url.openConnection();
// get a stream and read the certificate
InputStream is = con.getInputStream();
X509Certificate cert = new X509Certificate(is);
```

As you can see, the only difference between the two examples is that we have used an LDAP URL in the second case. However, for running the LDAP sample you first will have to register the LDAP protocol handler of IAIK-JCE to tell the `java.net` URL framework where to look for the LDAP supporting classes of IAIK-JCE:

```
System.getProperties().put("java.protocol.handler.pkgs",
                           "iaik.x509.net");
```

After having registered the IAIK LDAP protocol handler, an IAIK-JCE `LdapURLConnection` object is returned when calling `url.openConnection` for an LDAP URL.

Downloading certificates and crls from LDAP directories

We already have shown in the last chapter how to get a certificate from an LDAP directory. For downloading a certificate revocation list you only have to call `new X509CRL(is)` instead of `new X509Certificate(is)` to parse the crl from the stream connected to the LDAP server:

```
// register IAIK-JCE LDAP protocol handler
System.getProperties().put("java.protocol.handler.pkgs",
                           "iaik.x509.net");
// create LDAP URL
String urlStr = "ldap://ldap.signatur.rtr.at/"+
                "CN=Certificate%20Revocation%201,O=Telekom-Control-"+
                "Kommission,C=AT?certificateRevocationList;binary";
URL url = new URL("urlStr);
// open connection
URLConnection con = url.openConnection();
// get a stream and read the crl
InputStream is = con.getInputStream();
X509CRL crl = new X509CRL(is);
```

This example represents the most simple (and also most common) use case for downloading a crl from an LDAP directory. As typically for the CRLDistributionPoints extension of an X.509 certificate, all required query information is already contained in the URL string itself. The IAIK-JCE LDAP implementation also allows you to do more sophisticated LDAP search queries by configuring the `LdapURLConnection` with several request properties. You also may use it to do a full- or sub-tree search for retrieving all certificates/crls contained in an LDAP repository (sub-tree) that match some certain criteria (please refer to the IAIK-JCE Javadoc for a detailed description). IAIK-JCE also contains a command line utility for searching and downloading certificates and crls from LDAP repositories (see `cmd/ldapSearch` folder of the IAIK-JCE distribution).

Summary

This article shows how the IAIK-JCE `LdapURLConnection` implementation can be used to download certificates and crls from LDAP directories without knowledge about LDAP protocol or JNDI details.

References

1. Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile: http://www.ietf.org/rfc/rfc3280.txt
2. The LDAP URL format: http://www.ietf.org/rfc/rfc2255.txt
3. Java Naming And Directory Interface (JNDI): http://java.sun.com/products/jndi/
4. IAIK-JCE Toolkit: http://jce.iaik.tugraz.at/products/core_crypto_toolkits/jca_jce
5. Java-Source examples: CertificateDownload.java, CrlDownload.java