入門LDAP認証（3）
── 検索と認証（セキュア編）──

平　野　　靖

Ⅰ．はじめに

前回は，LDAPサーバ内の情報を検索する方法と認証する方法の基本を説明した。今回はより安全にLDAPサーバにアクセスする方法を説明する。LDAPサーバにアクセスするためには，IDとパスワードが必要であり，これが他人に知られると不正アクセスされる可能性がある。そのため，これらを他人に知られないようにすることが重要である。さらに，LDAPサーバにはさまざまな個人情報も含まれる。もしIDとパスワードを他人に知られないように対策をしても，通信路上を個人情報が暗号化されずに流れることには不安が残る。本稿では，通信のすべてが暗号化される方法としてSSL/TLSを紹介する。

Ⅱ．通信路の暗号化

1．公開鍵暗号

LDAP Version 3で規定されている暗号化通信の方法はSSL/TLSによるものであり，公開鍵暗号をベースにしている。公開鍵暗号は対になる2つの鍵（公開鍵と秘密鍵）を使って，盗聴・改竄・なりすまし・否認の防止が可能になる。詳細は文献［1］［2］などを参考にして欲しい。

情報連携基盤センターが用意したお試しLDAPサーバであるpub-ldap.itc.nagoya-u.ac.jpのサーバ証明書はお試しLDAPサーバ上に構築されたCA（認証局，Certificate Authority）によって署名されている。そのため，お試しLDAPサーバが本物であるかをクライアントが確認するためには，このCAの証明書が必要となる。この証明書はhttp://pub-auth-web.itc.nagoya-u.ac.jp/CA/cacert.pemに置いてあるので，必要な場合にはダウンロードして欲しい。

2．SSL/TLS

LDAPS

SSL（Secure Sockets Layer）［3］［4］はNetscape Communications社が開発した暗号化通信プロトコルである。SSLはLDAPに限らず，多くのアプリケーションで用いられている。例えば，httpsやssh，sftpなどもSSLを用いている。現在では，SSLの仕様決定はIETFのTLSワーキンググループに引き継がれ，名称はTLS（Transport Layer Security）と変更された。TLSの仕様は文献［5］［6］で定義されている。

SSL/TLSによって通信路を暗号化する方法のうちの1つはLDAPS（LDAP over SSL）と呼ば

れ，LDAPとは異なるポートを利用するものである。お試しサーバでは1025番ポートをLDAPSのために利用できるようにしてある。したがって，LDAPSのURIは下記のようになる。
ldaps://pub-ldap.itc.nagoya-u.ac.jp:1025

StartTLS拡張操作

LDAPSではLDAPサーバとクライアント間のすべての通信をSSL/TLSで暗号化することを前提としているが，StartTLS拡張操作（以下，単にStartTLSと言う）では，可能であれば，かつ任意の時点でSSL/TLSで暗号化通信を開始する。もし，暗号化通信ができない場合の挙動については，クライアントの設定に依存する。StartTLSで使うポートはLDAPと同じである。したがって，StartTLSでアクセスする場合のURIは下記のようになる。
ldap://pub-ldap.itc.nagoya-u.ac.jp:1024

３．公開鍵暗号プログラム

公開鍵・秘密鍵のペアを生成・変更・削除したり，証明書の管理をしたりするツールがいくつか公開されている。有名なものにOpenSSL［7］，Certificate Database Tool［8］，keytool［9］などがある。Netscape NavigatorやInternet Explorerなどでも，サーバ証明書を管理することができる。とくにNetscape Navigatorで生成された証明書データベースは，本特集の第１回で紹介したSofterraが提供するLDAP Browser［10］でも利用することができる。例えば，Netscape NavigatorでLDAPS用のポートをhttpsでアクセスする（https://pub-ldap.itc.nagoya-u.ac.jp:1025）ことにより証明書データベースを作ることができる。

JavaでLDAPクライアントを作る場合には，keytoolで証明書の格納用ファイル（キーストア）を作成する必要がある。詳細は付録を参照。

Ⅲ．ldapsearch

OpenLDAPのldapsearchでLDAPSによって接続するためには

```
% ldapsearch -H ldaps://pub-ldap.itc.nagoya-u.ac.jp:1025 \
  -b o=LDAP-TEST "(objectclass=*)"
```

と入力する。ただし，OpenLDAPがSSLをサポートするようにコンパイルされていなければならない。

`-H`オプションはLDAP URIを指定するためのもので，プロトコル（`ldaps`），ホスト名（`pub-ldap.itc.nagoya-u.ac.jp`），及びポート番号（`1025`）を一度に指定することができる。したがって，前回のように平文で通信する場合には，下記のように指定することもできる。

```
-H ldap://pub-ldap.itc.nagoya-u.ac.jp:1024
```

StartTLSを指定するためのオプションは `-Z` であり，指定すべきポートは1024番である。また，このオプションはStartTLSの指定であるため，StartTLSの開始が失敗した場合には暗号化されない状態で通信が行われる。一方，`-ZZ` オプションを指定すると，StartTLSの開始が成功しない限

り，通信は行われない。OpenLDAPのldapsearchでのStartTLSによるアクセスは下記のいずれかのオプションを付けて行われる（いずれも匿名認証の場合）。

```
% ldapsearch -Z -h pub-ldap.itc.nagoya-u.ac.jp -p 1024 \
  -b o=LDAP-TEST "(objectclass=*)"
% ldapsearch -ZZ -h pub-ldap.itc.nagoya-u.ac.jp -p 1024 \
  -b o=LDAP-TEST "(objectclass=*)"
```

なお，LDAPSによる場合も，StartTLSによる場合も，Solaris上で実行した際に

```
ldap_start_tls: Connect error (91)
        additional info: error:24064064:random number generator:
SSLEAY_RAND_BYTES:PRNG not seeded
```

というエラー（紙面の都合上，SSLEAY_RAND_BYTESの前で改行した）が出力されることがある。この場合には，下記のいずれかの対策を行う。

1. /etc/ssl/certsに証明書を置く。ただし，ディレクトリは環境によっては異なる可能性がある。
2. ldapsearchが証明書をチェックしないように下記のいずれかの対策を行う（本運用する場合には，推奨しない）。
   - 環境変数 LDAPTLS_REQCERT を never にセットする。例えば，csh 系のシェルの場合には，コマンドラインから
     ```
     % setenv LDAPTLS_REQCERT never
     ```
     と入力する。
   - /etc/ldap/ldap.conf に以下の設定を書く。ただし，ディレクトリは環境によっては異なる可能性がある。
     ```
     TLS_REQCERT never
     ```
   - ホームディレクトリに .rndというファイルを用意し，適当な内容を書き込んでおく。例えば，下記のようなコマンドを実行する。
     ```
     % ps -edalf > ~/.rnd
     ```

Ⅳ．サンプルプログラム

この章では，SSL/TLSによる通信のサンプルプログラムを示す。JNDIやJLDAPなどでSSL/TLS通信を行うためには，keytoolを用いてCA証明書を組み込み，クライアントがCA及びそのCAが発行したサーバ証明書を信頼する必要がある。CA証明書を組み込む方法は付録を参照して欲しい。さらに，http://java.sun.com/products/jsse/からJSSE（Java Secure Socket Extention）をダウンロードし，クライアントマシンにインストールする必要がある[1]。

下記のサンプルプログラムは，いずれも前回説明したプログラムのうち，主にSSL/TLS通信を行うときに変更すべき部分のみを掲載した。したがって，下記のプログラムをそのまま実行する

1　J2SEv1.4からは標準でJSSEが付属しているので，あらためてインストールする必要はない。

ことはできない。プログラムの全体はhttp://pub-auth-web.itc.nagoya-u.ac.jp/に置いてあるので，自由に使用してかまわないが，あくまでサンプルプログラムなので実際のサービスなどで使う場合には十分注意して欲しい。なお，サンプルプログラムの作成には文献［11］，及び文献［12］を，JSSEに関しては文献［13］を参考にした。

１．JNDI

LDAPS

まずは，JNDIでLDAPSによる通信を行う場合のサンプルプログラムを示す。LDAPSでの通信であるため，お試しサーバに対して指定するポートは1025番である。なお，注意すべき点は，JNDIでLDAP URIを指定する際には，“ ldaps ”ではなく，下記のように“ ldap ”を指定しなくてはならないことである。

```
env.put(Context.PROVIDER_URL, "ldap://" + host + ":" + port );
```

このプログラムを実行するには，

```
% javac ldapSearchSSL.java
```

でプログラムをコンパイルした後，例えば，

```
% java ldapSearchSSL pub-ldap.itc.nagoya-u.ac.jp 1025 o=LDAP-TEST \
  cn=DptStaff,ou=Staff,o=LDAP-TEST ps00002 "cn=*" ~/.keystore
```

と入力する。前回のプログラムとの違いは，最後にキーストアを指定する点である。なお，“ ~ ”はUNIX系OSでホームディレクトリをあらわす。もし別の場所にキーストアがある場合には，適宜変更すること。

―― ファイル名：ldapSearchSSL.java ――

```
import javax.naming.*;
import javax.naming.directory.*;

import java.util.Hashtable;
import java.util.Enumeration;

public class ldapSearchSSL {

   public static void main(String[] args) {

      if (args.length != 7 ){
         System.out.println(
            "usage: ldapSearch host port BaseDN BindDN BindPW filter Keystore");
         System.exit(0);
      }

      String host   = args[0];
```

```
        String port   = args[1];
        String BaseDN = args[2];
        String BindDN = args[3];
        String BindPW = args[4];
        String Filter = args[5];
        String Keystore = args[6];

        // セキュリティプロバイダとして JSSE を設定
        java.security.Security.addProvider(new com.sun.net.ssl.internal.ssl.Provider());

        // キーストアの指定
        System.setProperty("javax.net.ssl.trustStore", Keystore);

        Hashtable env = new Hashtable();
        env.put(Context.INITIAL_CONTEXT_FACTORY, "com.sun.jndi.ldap.LdapCtxFactory");

        // TLS 利用の指定
        env.put(Context.SECURITY_PROTOCOL, "ssl");

        env.put(Context.INITIAL_CONTEXT_FACTORY, "com.sun.jndi.ldap.LdapCtxFactory");
        env.put(Context.PROVIDER_URL, "ldap://" + host + ":" + port );
        env.put(javax.naming.Context.SECURITY_AUTHENTICATION, "simple");
        env.put(javax.naming.Context.SECURITY_PRINCIPAL, BindDN );
        env.put(javax.naming.Context.SECURITY_CREDENTIALS, BindPW);

        try {
           DirContext ctx = new InitialDirContext(env); // 初期コンテキスト作成

               （以下，省略）
                    .
                    .
                    .
}
```

―――*―――*―――*―――

StartTLS

つぎに，JNDIでStartTLSによる通信を行う場合のサンプルプログラムを示す。ファイル名がldapSearchStartTLS.javaであることを除けば，LDAPSの場合と同様にコンパイル，及び実行ができる。LDAP用のポートを用いるので，お試しLDAPサーバでは1024番ポートを指定する。

―― ファイル名：ldapSearchStartTLS.java ――

```
import javax.naming.*;
import javax.naming.directory.*;
import javax.naming.ldap.*;
import javax.net.ssl.*;

import java.util.Hashtable;
import java.util.Enumeration;
import java.io.IOException;

public class ldapSearchStartTLS {

    public static void main(String[] args) {

        if (args.length != 7 ){
            System.out.println(
                "usage: ldapSearch host port BaseDN BindDN BindPW filter Keystore");
            System.exit(0);
        }

        String host   = args[0];
        String port   = args[1];
        String BaseDN = args[2];
        String BindDN = args[3];
        String BindPW = args[4];
        String Filter = args[5];
        String Keystore = args[6];

        // セキュリティプロバイダとして JSSE を設定
        java.security.Security.addProvider(new com.sun.net.ssl.internal.ssl.Provider());

        // キーストアの指定
        System.setProperty("javax.net.ssl.trustStore", Keystore);

        Hashtable env = new Hashtable();
        env.put(Context.INITIAL_CONTEXT_FACTORY, "com.sun.jndi.ldap.LdapCtxFactory");
        env.put(Context.PROVIDER_URL, "ldap://" + host + ":" + port );
        env.put(javax.naming.Context.SECURITY_AUTHENTICATION, "simple");
        env.put(javax.naming.Context.SECURITY_PRINCIPAL, BindDN );
        env.put(javax.naming.Context.SECURITY_CREDENTIALS, BindPW);

        try {
            LdapContext ctx = new InitialLdapContext(env, null);
```

```
        // Perform a StartTLS extended operation
        StartTlsResponse tls =
           (StartTlsResponse) ctx.extendedOperation(new StartTlsRequest());

        try {
           SSLSession session = tls.negotiate();
        } catch(IOException e) {
           System.out.println("JNDI Error: "+ e.toString());
        }

            （以下，省略）
                 .
                 .
                 .
}
```

―― * ―― * ―― * ――

２．JLDAP

LDAPS

つぎに，JLDAPの場合の例を示す。JNDIの場合と同様に，セキュリティプロバイダとしてJSSEを設定し，キーストアを指定する。さらに，LDAPサーバへの接続時にLDAPS用のソケットを指定する。

―― ファイル名：ldapSearchSSL.java ――

```
import com.novell.ldap.LDAPAttribute;
import com.novell.ldap.LDAPAttributeSet;
import com.novell.ldap.LDAPConnection;
import com.novell.ldap.LDAPEntry;
import com.novell.ldap.LDAPException;
import com.novell.ldap.LDAPSearchResults;
import com.novell.ldap.util.Base64;
import java.util.Enumeration;
import java.util.Iterator;
import java.io.UnsupportedEncodingException;

public class ldapSearchSSL
{
   public static void main(String[] args)
   {

      if (args.length != 7 ){
```

```
            System.out.println(
                "usage: ldapSearch host port BaseDN BindDN BindPW filter Keystore");
            System.exit(0);
        }

        String host   = args[0];
        int    port   = Integer.parseInt(args[1]);
        String BaseDN = args[2];
        String BindDN = args[3];
        String BindPW = args[4];
        String Filter = args[5];
        String Keystore = args[6];

        // セキュリティプロバイダとして JSSE を設定
        java.security.Security.addProvider(new com.sun.net.ssl.internal.ssl.Provider());

        // キーストアの指定
        System.setProperty("javax.net.ssl.trustStore", Keystore);

        // LDAPS 用のソケットファクトリを指定してセッションを作成
        LDAPConnection ld =
            new LDAPConnection(new com.novell.ldap.LDAPJSSESecureSocketFactory());

                （以下，省略）
                    .
                    .
                    .
}
```

―― * ―― * ―― * ――

StartTLS

JLDAPでStartTLSによる接続を行う方法は，LDAPSで接続を行う場合とほとんど同じである。異なるのは `LDAPConnection` でLDAPS用のソケットの代わりに，StartTLS用のソケットを指定する点のみである。

―― ファイル名：ldapSearchStartTLS.java ――

```
import com.novell.ldap.LDAPAttribute;
import com.novell.ldap.LDAPAttributeSet;
import com.novell.ldap.LDAPConnection;
import com.novell.ldap.LDAPEntry;
```

```
import com.novell.ldap.LDAPException;
import com.novell.ldap.LDAPSearchResults;
import com.novell.ldap.util.Base64;
import java.util.Enumeration;
import java.util.Iterator;
import java.io.UnsupportedEncodingException;

public class ldapSearchStartTLS
{
   public static void main(String[] args)
{

       if (args.length != 7 ){
          System.out.println(
             "usage: ldapSearch host port BaseDN BindDN BindPW filter Keystore");
          System.exit(0);
       }

       String host   = args[0];
       int    port   = Integer.parseInt(args[1]);
       String BaseDN = args[2];
       String BindDN = args[3];
       String BindPW = args[4];
       String Filter = args[5];
       String Keystore = args[6];

       // セキュリティプロバイダとして JSSE を設定
       java.security.Security.addProvider(new com.sun.net.ssl.internal.ssl.Provider());

       // キーストアの指定
       System.setProperty("javax.net.ssl.trustStore", Keystore);

       // Start TLS 用のソケットファクトリを指定してセッションを作成
       LDAPConnection ld =
          new LDAPConnection(new com.novell.ldap.LDAPJSSEStartTLSFactory());

              （以下，省略）
                   .
                   .
                   .
}
```

―― * ―― * ―― * ――

３．PHP

PHPでSSL/TLS通信を行うのは非常に簡単である[2]。LDAPで通信する際のプログラムの関数の引数を変えるか，関数を１つ追加するだけでよい。以下，該当する部分のみを示す。なお，PHPでSSL/TLSを用いて通信するには，SSLサポートを指定してOpenLDAPをコンパイルし，さらにPHPをコンパイルする際にもSSLを指定しなくてはならない。

LDAPSで通信を行う場合には，LDAPサーバに接続する際に

```
$ld = ldap_connect("ldaps://".$host, $port);
```

のように，プロトコルとして`ldaps`を指定するだけでよい。なお，`$host`，及び`$port`という変数には，それぞれLDAPサーバのホスト名，及びLDAPS用のポート番号が代入されているものとする。

StartTLSを実現するためには，`ldap_start_tls($ld);`という関数を呼ぶ。この関数を実行した直後からSSL/TLSでの通信が始まる。なお，`$ld` は `ldap_connect($host, $port);`の戻り値である。`$host`にはLDAPサーバのホスト名が代入されているものとする。StartTLSの場合には，LDAPと同じポートを用いるので，`$port` には，LDAP用のポート番号を代入しておく。

Ⅴ．むすび

今回はSSL/TLSによるLDAPサーバへのアクセスのための２つの方法をサンプルプログラムとともに説明した。部局向け，あるいは全学向けに認証付きの情報サービスを提供している職員の方は参考にしていただき，情報連携基盤センターが運用しているLDAPサーバを活用していただきたい。

次回はLDAPサーバを認証システムとして利用するアプリケーションの例を示し，実際にLDAPサーバがどのように利用できるのかを説明する。

付録：keytoolによるCA証明書の格納

http://java.sun.com/products/jsse/からJava Secure Socket Extension（JSSE）をダウンロードし，インストールする。つぎに，keytool を実行するディレクトリにpub-ldap.itc.nagoya-u.ac.jpのCA証明書をダウンロードして，

```
% keytool -import -file cacert.pem -trustcacerts
```

と入力すると，`Enter keystore password:` というプロンプトが表示されるのでキーストアのパスワードを入力する。なお，キーストアを新規に作成する場合には，パスワードを決定して入力する。さらに，`Trust this certificate? [no]:` に“ yes ”と入力する。これによって，新規の場合には `$HOME/.keystore` が作成される。すでに `$HOME/.keystore` が存在する場合には更新される。なお，`$HOME` はホームディレクトリを意味する。

2　SSL/TLSのサポートはVer.4.0.4で追加された。

参考文献

［1］John Viega, Matt Messier, Pravir Chandra 著，齋藤孝道訳：“ OpenSSL－暗号・PKI・SSL/TLSライブラリの詳細－ ”，オーム社，東京，2004

［2］小松文子他：“ 改訂　PKIハンドブック ”，ソフトリサーチセンター，東京，2004

［3］http://wp.netscape.com/eng/ssl3/draft302.txt

［4］http://mars.elcom.nitech.ac.jp/Research/MM/security/ssl/draft302-j.html（文献［3］の和訳）

［5］http://www.ipa.go.jp/security/rfc/RFC2246-00EN.html

［6］http://www.ipa.go.jp/security/rfc/RFC2246-00JA.html（文献［5］の和訳）

［7］http://www.openssl.org/

［8］http://www.mozilla-japan.org/projects/security/pki/nss/tools/certutil.html

［9］http://java.sun.com/j2se/1.3/docs/tooldocs/win32/keytool.html

［10］http://www.ldapbrowser.com/

［11］稲地　稔：“ OpenLDAP入門－オープンソースではじめるディレクトリサービス－ ”，技術評論社，東京，2003

［12］http://java.sun.com/j2se/1.4/ja/docs/ja/api/javax/naming/ldap/package-summary.html

［13］http://java.sun.com/j2se/1.4/ja/docs/ja/guide/security/jsse/JSSERefGuide.html

（ひらの　やすし：名古屋大学情報連携基盤センター大規模計算支援環境研究部門）