



How to setup SSL Connection in Motorola Linux phones

October 3, 2005

WHITE PAPER

How to setup SSL Connection in Motorola Linux phones

By

Motocoder Staff

SSL is a frequently used protocol that allows clients and servers to communicate safely over none secure public networks. In this article, you'll learn the basic concepts in SSL, and know how to setup a SSL connection in Motorola Linux handsets.

Basic concepts

First, let's introduce several basic concepts in SSL.

SSL

SSL (Secure Socket Layer, also called Transport Layer Security, TLS) is a protocol over TCP/IP, which is used for making any TCP/IP connection secure. It offers a secure socket that like TCP/IP socket to higher level protocols. For example, Https is HTTP protocol running on a SSL socket. SSL uses certificates to proof who you are, and then setup a safe session with encryption. SSL API is used to provide a means for developers to safely and securely transporting private data between client and server.

Certificate

SSL uses a cryptographic system that uses two keys to encrypt data – a public key known to everyone and a private or secret key known only to the sender of the message. A certificate is a data segment used to identity the owner; it includes a public key, the certificate info and the information of the owner of that key. With certificate, you can make sure the site you're connecting is exactly what it claims to be.

Digital Signature

A digital signature is a kind of message data processing with the sender's private key that can be used to authenticate the identity of the sender of a message, and to ensure that the original content of the message or document that has been sent is unchanged.

A digital signature can be used with any kind of message, so that the receiver can be sure of the sender's identity and that the message arrived intact.

Certificate Authority (CA)

In order to make sure that no one can insert a false identity in the certificate, the certificate is digitally signed by a certificate authority (the certificate issuer, CA). The certificate also identifies the certificate issuer with the CA certificate. Thus, you can trust a certificate to the degree that you trust the certificate issuer.

Certificate Chain/Root CA

There are so many certificates, most of which can't be directly trusted when it is received from the server. So, a mechanism like chain proof is used.

When the message recipient gets a certificate, usually, what he really gets is a certificate chain like below:

A (message sender certificate) – (certified by)→
B (certificate of CA) – (certified by)→
C (certificate of higher level CA) ...– (certified by)→
Certificate of Root CA

A trust anchor is a CA certificate that you have installed in the device and acts as the anchor of your trust. If any of the trust anchors is included in the certificate chain, the certificate is trusted. A root CA certificate is a self-signed CA certificate. Usually, several root CA certificates will be installed in the handset as the anchor of the certificate chain.

Setup SSL Connection

Generally to say, there're three steps to setup a SSL connection: first, make SSL enabled in the server and put the server certificate in its keystore; next, import the server's root CA certificate in the handset as a trust anchor; last, the client application in the handset opens a ssl or https connection to the server. In our demo, the server's certificate is a self signed root certificate.

Environment of the demo application

Tomcat 5.5
J2SE 1.5_02
SSL port: 8443
IP address: 221.217.234.174 (Please modify to your own IP address or domain name)
Motorola Linux phones (A760, A768, A780, E680)

Setup SSL connection in Tomcat

Tomcat5.5 is selected in the server side. First, use J2SE keytool (<jdkpath>\bin\keytool.exe) to generate the server root certificate and keystore. (keystore is a small database with keys and certificates in it)

```
<keystorePath>:\>keytool -genkey -keyalg RSA -alias youraliasname -keystore sslDemokeystore
```

The application will ask for some information. Note: the IP address or the domain name may be input in the first entry (Your first and last name). Then input the certificate information, "testtest" is used as password in our tiny demo application.

Then modify the server.xml in Tomcat5.5\conf\, remove the comments tags, make SSL enabled and add the keystore info.

```
<Connector port="8443" maxHttpHeaderSize="8192"  
    maxThreads="150" minSpareThreads="25" maxSpareThreads="75"  
    enableLookups="false" disableUploadTimeout="true"  
    acceptCount="100" scheme="https" secure="true"  
    clientAuth="false" sslProtocol="TLS" keystoreFile="sslDemokeystore"  
    keystorePass="testtest"  
>
```

The server certificate needs to be taken out from the keystore so as to import to the handset as a trust anchor. To export the certificate in keystore:

```
<keystorePath>:\>keytool -export -alias youraliasname -file sslDemo.cer -keystore sslDemokeystore
```

At last, a servlet called SSLDemoServlet used for send response is written and deployed.

Set certificate in Linux handsets

In the client side, the first step to setup a SSL connection is authentication. There're two kinds of authenticate in SSL: Server authentication, which means the server offer its certificate to the client; client authentication, which refers to the client give its certificate to the server. In Linux phones, only server authentication is implemented. To make the server certificate being trusted, you must import the root CA certificate of the server in the handset's cert manager as trust anchor.

All Motorola Linux series phones (A760, A768, E680, A780, E680i) support SSL and WTLS certificate. These certificates in DER encoded binary X509(.cer) format can be installed from cert manager as trust anchors. For A760/A768/E680/A780, the cert manager can be accessed in the wap browser. To insert a certificate as trust anchor, in the wap browser's menu select security -> Root certificate -> SSL certificates -> insert then import the server's root CA certificate (sslDemo.cer in our demo) .

Set up SSL connection in J2ME midlet

Only SSL certificate is supported in Java. To setup a SSL (or Https) connection in midlet:

```
// SSL connection
co = (Connection) Connector.open(ssl://... :8443);

//Https connection & print certificate infomation
hc =
(HttpsURLConnection)Connector.open(https://... :8443//SSLDemoServer/servlet/SSLDemoServlet);
info = hc.getSecurityInfo();
c = info.getServerCertificate();
String name = c.getIssuer();
testResult = "Test Success, certificat issuer:" + name ;
```

If there's an exception like:

"javax.microedition.pki.CertificateException: Certificate was issued by an unrecognized entity".

That means the certificate has not been imported properly, or the handset could not recognize the certificate. Please check it in the handset's cert manager.

Conclusion

Now you have seen how to setup a SSL or Https connection in Motorola Linux handsets. But it is also important to remember: SSL is only a part of the security system, it only secure the data transmitting and it has much higher computing resource requirement than none secure data transmitting.

For more information

TLS protocol version 1.0 as defined in <http://www.ietf.org/rfc/rfc2246.txt>

SSL protocol version 3.0 as defined in <http://home.netscape.com/eng/ssl3/draft302.txt>