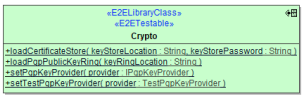
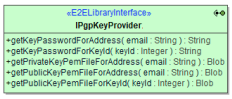
Keys and Certificates

For some functionality the library needs to know about certificates and keys:

- **Verification of secure connections**
  To verify secure connections (SSL/TLS/HTTPS) to mail servers, the underlying JVM must know the certificate chain. If you get errors indicating that the SSL handshake fails due to certificate errors, most likely the JVM can not verify the certificate sent by the mail host.
  You can fix this by either
  - importing the certificates (or all missing certificates of that certificate chain) into the JVM's keystore so they can be found automatically by the JVM.
  - importing them into a separate Java Key Store (JKS). **Only in this case** you need to load this separate keystore by means of loadCertificateStore().
- **Verification of PGP signatures**
  To verify PGP signatures, the public key of signers must be known. To do this, you can load a PGP public key ring into the library. Refer to you PGP installation about how to export public keys (this will be replaced in future versions by the IPgpKeyProvider callback, see next item)
- **Encryption and decryption of messages**
  To encrypt or decrypt OpenPGP messages, the public keys of addressees (encryption), the private key of the receiver (decryption), or the public key of the sender (signing) must be provided. For this purpose the library defines an interface IPgpKeyProvider, and the operation setPgpKeyProvider(provider). So the actual implementation of key handling is left to the model importing the library: design a class that implements this interface and register it using the Crypto::setPgpKeyProvider(your implemenation) before calling any operations that may ask for keys.





On this Page:

Related Pages:

- Mail Server Connection
- JavaMail Library Reference

IPgpKeyProvider Interface

At certain points in the process of encrypting or decrypting messages, public and/or private PGP key are required. In case of private keys, the respective password to access the key is also required. The interface defines the following operations that are called during the process on demand.

When sending emails, keys and passwords are retrieved by the respective email addresses used when sending the mail (FROM, TO, CC, BCC). When receiving emails, required keys are referenced within an email by their key id. Hence keys/passwords may be retrieved by email address and/or key id.

- retrieving the key itself, expected result is the binary content of the key in PEM format, or NULL when no such key exists
  - getPublicKeyPemFileForAddress(email): called when encrypting for an addressee
  - getPublicKeyPemFileForKeyId(keyId): called when verifying senders' signatures
  - getPrivateKeyPemFileForAddress(email): called when signing an email
- retrieving the password for private keys
  - getKeyPasswordForAddress(email): called when signing an email
  - getKeyPasswordForKeyId(keyId): called when decrypting an email

PGP Key IDs

PGP Keys carry some details like type, validity, creation and expiry date, fingerprint, email address, and a "Key ID" which is typically displayed as a string in hexadecimal notation, e.g. '048F841A'. Typically this is also part of the filename when a key is exported from key management software in PEM format, and it can be accessed from the UI of your key management software or using the command line tool of choice. The callback operations will ask for this Key ID, **but by its decimal integer value**, in this example 76514330. Future versions might offer to retrieve the keys by the hexadecimal key value as well.

Crypto Operations

loadCertificateStore

Parameter	Types	Direction	Description	Allowed Values / Example
keyStoreLocation	String	in	Specify the path to the keystore file.	`opt/bridge/crypto/mailservercerts.jks`

keyStorePassword	String	in	Specify the password of the keystore.	

loadPgpPublicKeyRing

> **to be discontinued**
>
> Future versions of the library will rely on setPgpKeyProvider() solely (see below), this operation will disappear. Until then it is still required if you want to verify signatures of unencrypted emails.

Parameter	Types	Direction	Description	Allowed Values / Example
keyRingLocation	String	in	Specify the path to the keyring file.	`opt/bridge/crypto/pubkeys.asc`

setPgpKeyProvider

Parameter	Types	Direction	Description	Allowed Values / Example
provider	IPgpKeyProvider	in	Instance of a class implementing the IPgpKeyProvider interface	