IAIK-JCE

IAIK-JCE is a set of APIs and implementations of cryptographic functions including digest, signature, message authentication code, key agreement, symmetric, asymmetric, stream and block encryption methods. It supplements the security functionality of the default JDK (JCA/JCE) and extends it about additional features and crypto protocol implementations. It offers a wide range of cryptographic algorithms (e.g. RIPEMD, SHA family, SHA-3, HMAC, DSA, RSA, DH, DES, 3DES, AES, etc.) which can be used via the standard JCA/JCE API.
The X.509 package supports creation and parsing of X.509 public key, qualified and attribute certificates. Revocation information can be provided and verified via CRLs and OCSP. LDAP utilities maybe used to search LDAP directories for certificates and attribute certificates or download CRLs from their distribution points.
Comprehensive ASN.1 and PKCS APIs allow modelling of ASN.1 structures, secure storing of sensitive keying and data material, and signing or encrypting of digital documents.
The cryptographic functionality of IAIK-JCE is extended with elliptic curve cryptography and smartcard and HSM support by means of the IAIK-ECCelerate and IAIK-PKCS#11 toolkits. A particular lightweight version (IAIK-JCE ME) is optimized for usage with small and mobile devices.

Main Benefits

- **Easy to use**: IAIK-JCE comes with an easy-to-use and intuitive API. The comprehensive Javadoc documentation and rich set of demo samples helps you to get started quickly and soon become familiar with cryptography for the Java^{TM} platform and IAIK-JCE.
- **Performance**: IAIK-JCE has been optimized for speed and memory consumption. For our algorithm implementations we have utilized all our programming knowledge and experience to write effective code and get a maximum of speed. Our PKCS#7 library and X.509 CRL implementations provide stream-based versions which allow the signing/encrypting of large amounts of data or parse big revocation lists without running into memory problems.
  IAIK-JCE (version CC Core 3.1) has passed a Common Criteria evaluation with level EAL 3+. This gives you the assurance that an

independent professional security evaluation body has inspected our toolkit (which is especially interesting for the use in financial and e-business applications).

- **Interoperability**: IAIK-JCE provides the cryptographic basis for all IAIK crypto toolkits implementing standard protocols like S/MIME, SSL/TLS, or XML Signature and XML Encryption. Extensive testing and usage in real-world applications for a long time guarantee a maximum level on interoperability.
- **Extensibility and Flexibility:** Where possible, IAIK-JCE can easily be extended by user-written code. The X.509 package, for instance, allows it to plug-in private certificate extensions in a straightforward manner. Our PKCS#7 library makes it possible to integrate smartcard operations in an easy way. This design principle is maintained and further developed by the IAIK-JCE extending SSL/TLS, CMS and S/MIME toolkits.

Feature List

- Written entirely in the Java™ language
- Support for JDK Versions 1.2, 1.3, 1.4, 1.5, 1.6, 1.7, 1.8 and compatible
- Reliability and robustness through use in real world applications over many years
- Developed and maintained by experts in applied cryptography and the Java™ programming language
- Comprehensive documentation and many demos
- Hash algorithms: SHA3-224, SHA3-256, SHA3-384, SHA3-512, SHA-1, SHA-224, SHA-256, SHA-384, SHA-512, SHA-512/224, SHA-512/256, MD2, MD5, RIPEMD-128, RIPEMD-160, RIPEMD-256, RIPEMD-320, WHIRLPOOL, GOST-3411
- SHA3 Extendable Output Functions (XOFs):
  SHAKE128 (SHAKE128InputStream), SHAKE256 (SHAKE256InputStream)

- SHA3 candidate hash algorithms:
  BLAKE-224, BLAKE-256, BLAKE-384, and BLAKE-512,
  Groestl-224, Groestl-256, Groestl-384, and Groestl-512,
  JH-224, JH-256, JH-384, and JH-512,
  KECCAK-224, KECCAK-256, KECCAK-384, and KECCAK-512, as well as
  Skein-224, Skein-256, Skein-384, and Skein-512

- Signature schemes
  - PKCS#1 version 1.5 RSA with SHA-1, SHA-224, SHA-256, SHA-384, SHA-512, SHA512/224, SHA512/256, SHA3-224, SHA3-256, SHA3-384, SHA3-512, MD2, MD5, RIPEMD-128, RIPEMD-160, RIPEMD-256, WHIRLPOOL
  - PKCS#1 version 2.1 RSA PSS with SHA-1, SHA-224, SHA-256, SHA-384, SHA-512, SHA512/224, SHA512/256, SHA3-224, SHA3-256, SHA3-384, SHA3-512, MD2, MD5, RIPEMD-128, RIPEMD-160, WHIRLPOOL;
    Support for RSASSA-PSS keys according to RFC 4055
  - ISO 9796-2 (2002) RSA with SHA-1, SHA-256, SHA-384, SHA-512, RIPEMD-128, RIPEMD-160, WHIRLPOOL;
  - Raw RSA with external hashing
  - SSL/TLS RSA signature with MD5 and SHA-1
  - DSA with SHA-1, SHA-224, SHA-256, SHA-384, SHA-512, SHA3-224, SHA3-256, SHA3-384, SHA3-512;
  - Raw DSA with external hashing

- Symmetric ciphers AES, DES, Triple-DES (112 and 168 bit), IDEA, Blowfish, Camellia, GOST, CAST-128, RC2, ARCFOUR (compatible with RC4^{TM}), ChaCha20, RC5, RC6, password-based (PKCS#5 PBES1 with MD5, SHA-1 and DES, Triple-DES, RC2; PKCS#5 PBES2 with AES, DESede, … and HMAC/SHA-1, HMAC/SHA-2), MARS, Twofish, Rijndael, Rijnael-256, Serpent, AES-CBC-CMAC
  - key generation
  - Password based key derivation (PKCS#5, PBKDF2; PKCS#12)
  - encryption and decryption
  - ECB, CBC, PCBC, CFB, OFB, OpenPGPCFB, CTR, CCM, GCM, CTS modes of operation
  - padding PKCS#5, SSL version 3.0, ISO 7816-4, ISO 10126-2,

no padding

- Optional AES addon which makes use of the AES-NI instruction set extensions of modern x86 CPUs

- Key-Wrap ciphers for AES, Camellia, Triple-DES, CAST-128, RC2, HMAC-AES, HMAC-Triple-DES

- RSA encryption/decryption according to PKCS#1 version 1.5 and 2.1 OAEP; Support for RSAES-OAEP keys according to RFC 4055

- Blinding for RSA private key operations

- ElGamal encryption/ decryption according to PKCS#1 version 1.5

- HMACs with SHA-1, SHA-224, SHA-256, SHA-384, SHA-512, SHA512/224, SHA512/256, SHA3-224, SHA3-256, SHA3-384, SHA3-512, MD5, RIPEMD-128, RIPEMD-160, WHIRLPOOL, GOST-3411

- CMAC with AES and Triple-DES
- CBC-MAC with AES, DESede, DES according to ISO/IEC 9797-1

- Key agreement with DH (Diffie Hellman) and ESDH (Ephemeral Static DH)

- Key-Pair generation for RSA (unlimited key-size) according to IEEE P1363, Key-Pair generation for RSA according to FIPS 186-3, RSASSA- PSS, RSAES-OAEP, DSA, DSA-SHA2, DH, ESDH

- Random
    - NIST SP800-90 with SHA-1, SHA-224, SHA-256, SHA-384, SHA-512, HMAC/SHA-1, HMAC/SHA-224, HMAC/SHA256, HMAC/SHA-384, HMAC/SHA-512, AES-128, AES-192 and AES-256
    - FIPS 186 with SHA-1, SHA-224, SHA-256, SHA-384, SHA-512, RIPEMD-160
    - ANSI X9.17
    - BSI AIS 20 (v2.0) E.5 with SHA-1, SHA-224, SHA-256, SHA-384, SHA-512, MD5, RIPEMD-128, RIPEMD-160, WHIRLPOOL

- Software key stores
  - PKCS#12
  - Proprietary (IAIKKeyStore)

- ASN.1 encoding and decoding (DER, BER, Base 64)

- X.509 Certificates
  - public key and attribute certificates
  - encoding and decoding
  - parsing and creation (issuing) of certificates
  - PKCS#7 and X.509 certificate lists
  - a rich set of predefined extensions (PKIX, Netscape and more)
  - support for custom extensions
  - qualified certificates
  - implementations of all mandatory attributes of the PKIX AttributeCertificate profile and more
  - support for custom attributes

- X.509 CRLs with support for indirect and delta CRLs

- Stream based X.509 CRL implementation to parse large CRLs for revocation information without memory problems

- OCSP (Online Certificate Status Protocol), client side and server side
  - direct support for transport via TCP and HTTP
  - OCSP response creation based on CRLs

- LdapURLConnection implementation allowing to search LDAP directories for certificates attribute certificates or CRLs as accustomed from the java.net URL framework

- Password Generator, Password Store and Password Strengh Checker utilities

- PKCS support
  - PKCS#1 version 1.5 signature and encryption
  - PKCS#1 version 2.1 PSS signatures and OAEP encryption
  - PKCS#5 password-based cryptography
  - PKCS#7 with support for single-pass stream processing

- PKCS#8 private key information syntax
- PKCS#9 selected object classes and attribute types
- PKCS#10 certificate requests
- PKCS#12 personal information exchange syntax (private keys)