

Analysis of capabilities and performance of JAVA Cryptography Extension (JCE)

Vani Karimijji

Specification:

Introduction

Implementations of all security protocols require usage of tools that allow encryption and decryption to achieve the desired security. Hence it is very important to understand the performance of the tools that are used in this context to be able to analyze their adaptability to different protocols and domains.

The Java API, *JCE* (Java Cryptography Extension) part of J2SE SDK v1.4, defines the general architecture and specific services for cryptographic operations. The Java Cryptography Extension (JCE) provides a framework and implementations for encryption, key generation and key agreement, and Message Authentication Code (MAC) algorithms. Support for encryption includes symmetric, asymmetric, block, and stream ciphers. The software also supports secure streams and sealed objects.

The JCE API (Application Programming Interface) covers:

- Symmetric bulk encryption, such as DES, RC2, AES and IDEA
- Symmetric stream encryption, such as RC4
- Asymmetric encryption, such as RSA with key sizes 1024, 2048, 4096
- Password-based encryption (PBE)
- Key Agreement schemes: Diffie-Hellman (DH)
- Message Authentication Codes (MAC): MD5-MAC, HMAC, XOR-MAC, CBC-MAC, and DMAC.

The purpose of this project is to compare the performance of software implementations of the above algorithms in JCE with their corresponding implementations in C/C++ and assembly language.

Language, platform, and compiler used for a primary implementation (e.g., gnu C under Unix). Additional software required.

The experiments will be performed on Windows XP over a 1.79 GHz Pentium processor.

The compilers are:

1. Java 2 SDK, Standard Edition Version 1.4.1 using javax.crypto package
2. C and C++ compilers from GCC using Crypto++ library

Any additional software required for experimentation on large numbers will be decided later as the project progresses.

Specification of the input and output of the program(s).

For encryption:

Input file will be a plain text to a cipher object and an input key of specified length to the cipher object for encryption. Output will be a cipher text.

For decryption:

Input file will be a cipher text to the cipher object and an input key. Output will be plain text.

Algorithms considered for performance analysis

DES, RC2, AES, Twofish, IDEA, RC4, RSA (1024, 2048), MD5-MAC, HMAC, XOR-MAC, CBC-MAC, DMAC, Diffie-Hellman.

Procedures for testing the functionality and performance of the program(s). And Plan of Experiments to be performed

Implementations of algorithms in Java, C/C++, and ASM languages are compared for their execution times.

Input is provided via an input txt file and the program output is written to a file.

Execution time is compared for following cases

Time taken by the program with file I/O

Time taken by the program without file I/O

List of possible areas, where the specification can change depending on the progress of the project

Some of the Algorithms considered might have to be substituted if an implementation for them is not available in any of the considered languages

Time schedule

October 2nd: submit the final specification.

October 17th: study all the algorithms and check their implementations in all the languages, download all the software implementations of the algorithms

November 14th: complete coding part

December 5th: complete the testing part

December 12th: submit the project report

December 19th: Final oral presentation

List of literature:

Java Cryptography Extension (JCE) Reference Guide for the Java™ 2 SDK, Standard Edition, v1.4

<http://java.sun.com/j2se/1.4.2/docs/guide/security/jce/JCERefGuide.html#HmacEx>

Cryptography with java

<http://www.informit.com/articles/article.asp?p=170967&seqNum=4>

Inside Java 2 Platform Security: Architecture, API Design, and Implementation
By Li Gong, Gary Ellison, Mary Dageforde

<http://print.google.com/print?id=XfWlYWVzo20C&oi=fnd&pg=PA1&sig=tF-CVr5atiBcUXd5POMe9NATqWU>

Crypto ++ library

<http://www.eskimo.com/~weidai/cryptlib.html>

Java Cryptography Extension (JCE) by Sun Developer Network

<http://java.sun.com/developer/technicalArticles/Security/JCE/>
<http://java.sun.com/products/jce/index-122.html>

Foundations of cryptography:

<http://www.securitydocs.com/library/3547>

Common Cryptographic Algorithms

http://www.unix.org.ua/oreilly/networking/puis/ch06_04.htm

Security Protocols:

http://engr.smu.edu/cse/hacnet/Security_Protocols.htm

