skytechnologies

SkyMobile Security Whitepaper

Sky Technologies Pty Ltd
Suite 8, 758 Blackburn Rd,
Clayton Nth.
Australia. 3168

POWERED BY SAP NetWeaver™

skytechnologies

Contents

Introduction

This paper aims to outline the measures available within SkyMobile for securing data within an application. This is an important consideration in scenarios where (for example) confidential or sensitive data may be used or manipulated by an application running on a mobile device. It is essential to ensure that this type of data is not compromised, either during transmission or through loss or theft of the device itself.

The paper assumes a certain familiarity both with SkyMobile, and with network security and encryption techniques.

A (very) Brief Introduction to Encryption

There are really two major forms of encryption in use today: symmetric and asymmetric.

In symmetric encryption, the same key is used for both encryption and decryption. Its advantage is that it is very fast, but the fact that a single key is used tends to reduce its utility because control over the key becomes an issue.

Examples of symmetric encryption algorithms include DES, Triple DES, and Blowfish. There are many others.

Asymmetric encryption works in a slightly different way. A pair of matched keys (generally termed public and private) are used. Each key will decrypt data encrypted with the other. Asymmetric encryption is useful from a key management perspective, but is computationally intensive and hence very slow.

The best-known example of an asymmetric encryption algorithm is RSA.

Encryption of Communications

The SkyMobile deployment model fully supports encryption of transmitted data. There are two points at which a transmitted data "message" may need to be encrypted. These are:

- Data exchanged between the SkyMobile Java server and the XML gateway.
- Data exchanged between the presentation client and the SkyMobile Java server.

SkyMobile uses a combination of both symmetric and asymmetric encryption in order to achieve data security. This is a common technique that is present in protocols such as SSL, the standard library used by most browsers to secure data. It makes best use of the strengths of both symmetric and asymmetric encryption.

The technique involves an initial "handshake" that is encrypted via an asymmetric algorithm such as RSA. This initial exchange is used to agree a randomly chosen "session key". Once the session key has been generated and made known to partners, communications then switch to using a symmetric encryption algorithm such as Blowfish. The randomly chosen session key is used to symmetrically encrypt and decrypt the messages.

This approach ensures that only the two partners involved in the data exchange can know the session key, eliminating the key management issues associated with using symmetric encryption. It also doesn't suffer from the performance problems inherent in use of asymmetric encryption, because only the initial "handshake" is encrypted in this fashion. Hence, it is secure from a key management perspective, without being too slow.

The encryption features of all Sky products (including SkyMobile) are modular. This means that different algorithms can be "plugged in" depending on requirements. The current algorithms of choice are RSA for asymmetric encryption and Blowfish for symmetric encryption. Both of these algorithms are well-known and in the public domain.

Security of Stored Data

In any application that may potentially need to be available offline (i.e. when there is no connection back to the SAP server), there arises a need to locally persist stored data. The SkyMobile Java server makes use of a local database (often known as the LDB) that contains information about the application being invoked, as well as synchronized data downloaded from the SAP system. In cases where this data is sensitive or confidential, it is important to be able to secure it properly.

The database recommended for use in this type of scenario is SkyDB. SkyDB is an embedded relational database that is inbuilt in all Sky Java products. It permits data to be encrypted using a configured algorithm and key. Due to performance constraints, only symmetric encryption is supported, as asymmetric encryption is far too slow for this purpose.

Like the encryption of data communications, encryption of stored data is modular. Typically, the Blowfish algorithm is used. The symmetric key for the encryption and decryption of data is contained within the configuration file for the SkyMobile Java server itself. In order to prevent this key from being accessed, the configuration file itself can be encrypted as well. Encryption and decryption of the configuration file uses a key known only to Sky Technologies.

The Blowfish Encryption Algorithm

Overview

Blowfish is a public domain block cypher developed by a well-known security specialist named Bruce Schneier.

Description of the Blowfish Algorithm

Blowfish uses variable-length keys, up to a maximum size of 56 bytes (448 bits). In fact, even longer keys than this are possible (the key is segmented into 56-byte pieces and used in a round-robin fashion on consecutive data blocks). Obviously, with a variable length key, the longer the key, the more secure the communications are against a brute-force attack. By way of comparison, the industrial-strength version of SSL (the Secure Sockets Layer used by most browsers) uses 128-bit encryption. 448-bit encryption is 2 to the power of 320 times more resistant to a brute-force attack than this.

Performance

One further advantage of Blowfish is that it is very fast. Experience has shown us that using a 56-byte encryption key results in a negligible performance degradation when compared to unencrypted communications.

Sources & Further Reading

http://en.wikipedia.org/wiki/Blowfish_%28cipher%29

http://www.schneier.com/blowfish.html

The RSA Encryption Algorithm

Overview

The RSA encryption algorithm was developed by Ron Rivest, Adi Shamir, and Leonard Adleman at MIT in 1977. The RSA acronym represents the initials of the surnames of these three developers. It was the first public key cryptographic algorithm to be developed, and represented a very major advance in cryptography generally. It has stood the test of time, and is still very widely used worldwide, nearly 3 decades later.

Description of the RSA Algorithm

RSA involves a public and a private key. The public key can be known to everyone and is used for encrypting messages. Messages encrypted with the public key can only be decrypted using the private key.

RSA is based upon the mathematics of very large prime numbers. Put simply, it involves deriving a product for two very large prime numbers. This number will have only two factors, which represent the public and private key. The aim is to make the process of factoring the product number *computationally infeasible* by ensuring that it is sufficiently large. In other words, it must take much longer on average to factor the number via a brute-force attack than anyone would be willing to wait.

The threshold of what is computationally infeasible has been pushed back further and further as technology has advanced. Hence, selection of sufficiently long keys is very important when using this algorithm.

Performance

The RSA algorithm is computationally demanding, hence the need to combine it with other less resource-intensive algorithms in many applications.

Sources & Further Reading

http://en.wikipedia.org/wiki/Rsa

http://www.di-mgt.com.au/rsa_alg.html