

Mobile Certificate Application Specification (Objective-C Version)

Copyright: All right reserved SecuTech Solution, inc.

Table of Content

1. Relative Convention	1
1.1 Character Coding	1
1.2 Return Value	1
1.3 Public Key Format	1
1.4 Hash Algorithm Identification	1
1.5 Ease of Use Convention	1
1.6 Third Party Encrypt Library.....	2
1.7 Relative Development.....	2
1.8 Development Tools	2
2 Interface Definitation	2
2.1 Generate a Key-pair	2
2.2 Import a Public Key Certificate	3
2.3 Decrypt Digital Envelope	3
2.4 Digital Sign (including explicit key signature, ordinary signature, SSL handling, and P10 signature).....	4
2.5 Change a Device Password	5
2.6 Get DER Code of Public Key Certificate	5
2.7 Get the Remaining Number of Password Attempts	5
2.8 Check a Device Connection Status	2
2.9 Get a Device ID (the unique ID of a device).....	6
2.10 Error Code Table	6

1. Relative Convention

1.1 Character Coding

All string parameters are using utf-8 coding standard.

1.2 Return Value

All return values from functions are long. 0 means success, and <0 indicates an error code.

1.3 Public Key Format

pPublicKey is compatible with RFC 3447 specification.

```
RSAPublicKey ::= SEQUENCE {  
    modulus             INTEGER  -- n  
    publicExponent      INTEGER  -- e  
}
```

1.4 Hash Algorithm Identification

pHashOID is the identification string of each hash algorithm, and the specific algorithm ID are as the following table.

ID	Specification
1.3.14.3.2.26	SHA1 Algorithm
2.16.840.1.101.3.4.2.1	SHA256 Algorithm

1.5 Ease of Use Convention

As the speed of reading a certificate is low, an initiation mechanism shall be established, that is after the content of a public key is read at the first time of inserting a key, the public key and the information of the corresponding private key, such as container name, container ID and etc, shall be stored together in a configure file (the configure file will be created automatically: such as

“company name.config”) to ensure that at the next time, the public key and private key-pair can be established by just reading the private ID in the key. Through this method, the speed of certificate reading will be improved, and meanwhile the private key can be found very fast when signing or decrypting digital envelope.

1.6 Third Party Encrypt Library

If using encrypt library, please use Openssl(openssl1.0.0e) library file compiled by SecuTech.

1.7 Relative Development

Add “Company_product ID” into the name of code file in OC; add “Company_product ID” into the name of C and C++ source code file, function, and global variable to avoid conflict with other companies’ products in the upper compiled.

Realize the main function of the protocol function, and if it has a return value which is released automatically (not alloc), retain must be added.

1.8 Development Tools

- XCODE tools version 4.0 or higher;
- System default LLVM compiler 3.0;
- Support iOS SDK version 4.2 or higher (iOS deployment Target: iOS 4.2);
- Active Architecture: arm v6.

2 Interface Definitation

2.1 Check a Device Connection Status

-(long) isConnected:(bool *) pConnected;

Parameters:

[out] pConnected	If device is connected or not
------------------	-------------------------------

Returns:

S_OK	Success
<0	Fail, return error code

2.2 Generate a Key-pair

```
-(long) createRSAKey:(NSString *) pPasswordpublicKeyLength:( NSInteger)publicKeyLength
                    ppPublicKey:(NSData **) ppPublicKey
```

Parameters:

[in]	pPassword	Device password
[in]	publicKeyLength	Length of public key (bit), 1024 or 2048
[out]	ppPublicKey	Public key of the generated key-pair (format comply with the coding specification described in this document)

Returns:

S_OK	Success
<0	Fail, return error code.

2.3 Import a Public Key Certificate

```
-(long)importX509Certificate:(NSData *) pCertificate;
```

Parameters:

[in]	pCertificate	Der coded X509 public key certificate
------	--------------	---------------------------------------

Returns:

S_OK	Success
<0	Fail, return error code.

2.4 Decrypt Digital Envelope

```
-(long)decryptEnvelopeData:(NSData *) pCertificate
    pPassword:(NSString *) pPassword
    inData:( NSData *) inData
```

```
ppOutData:(NSData**)ppOutData;
```

Parameters:

[in]	pCertificate	Public key certificate corresponding to the private key, which is the decrypting certificate
[in]	pPassword	Device password
[in]	inData	Cipher text of digital envelope
[out]	ppOutData	Original text of digital envelope

Returns:

S_OK	Success
<0	Fail, return error code.

2.5 Digital Sign (including explicit key signature, ordinary signature, SSL handling, and P10 signature)

```
-(long)signData:(NSData *) pCertificate
```

```
pPassword:(NSString *) pPassword 私钥对应的公钥证书
```

```
srcData:( NSData *) srcData
```

```
pHashOID:( NSString*) pHashOID
```

```
ppOutData:(NSData**)ppOutData;
```

Parameters:

[in]	pCertificate	Public key certificate corresponding to private key, and if RSA key pairs are just generated, use the public key value returned from createRSAkey function.
[in]	pPassword	Device password
[in]	srcData	Source data to be signed
[in]	pHashOID	Hash algorithm identification
[out]	ppOutData	Signed data(PKCS1 format, without adding complementing bits)

Returns:

S_OK	Success
<0	Fail, return error code

2.6 Change a Device Password

```
-(long ) changePassword:(NSData *)pCertificate oldPwd:(NSString *) oldPwd newPwd:(NSString *)newPwd;
```

Parameters:

[in]	pCertificate	Public key certificate, and if it's device password, ignore this parameters.
[in]	oldPwd	Old password
[in]	newPwd	New password

Returns:

S_OK Success
 <0 Fail, return error code

2.7 Get DER Code of Public Key Certificate

```
-(long ) getX509Certificates:(NSArray **) ppCertificates;
```

Parameters:

[out]	ppCertificates	Public key certificate DER coded NSData array
-------	----------------	---

Returns:

S_OK Success
 <0 Fail, return error code

2.8 Get the Remaining Number of Password Attempts

```
-(long ) getPwdCanRetries:(NSData *)pCertificate pRemaining :(int *) pRemaining;
```

Parameters:

[in]	pCertificate	Public key certificate, and if it's the device password, ignore this parameter.
[out]	pRemaining	The remaining number of password attempts

Returns:

S_OK Success

<0 Fail, return error code

2.9 Get a Device ID (the unique ID of a device)

```
-(long ) getDeviceSerialNumber:(NSString **)ppDeviceSN;
```

Parameters:

[out] ppDeviceSN Device ID

Returns:

S_OK Success

<0 Fail, return error code

2.10 Error Code Table

Error Code	Macro definition	Remark
0	S_OK	Success
-1	S_DEVICE_FAILURE	No connection
-2	S_INVALID_PASSWORD	Verify password failed
-3	S_DEVICE_LOCK	Device Locked
-4	S_NOT_ENOUGH_MEMORY	No enough spaces of Memory
-5	S_INVALID_CERTIFICATE	Invalid format of Certificate
-6	S_INVALID_ALGORITHM	Invalid algorithm
-7	S_INVALID_PARAMETER	Invalid parameter
-8	S_UNKOWN_ERROR	Unknown error
-9	S_USER_CANCEL	Canceled by user