
Apple Certificate Library Functional Specification



2005-01-13



Apple Computer, Inc.

© 2005 Apple Computer, Inc.

All rights reserved.

No part of this publication may be reproduced, stored in a retrieval system, or transmitted, in any form or by any means, mechanical, electronic, photocopying, recording, or otherwise, without prior written permission of Apple Computer, Inc., with the following exceptions: Any person is hereby authorized to store documentation on a single computer for personal use only and to print copies of documentation for personal use provided that the documentation contains Apple's copyright notice.

The Apple logo is a trademark of Apple Computer, Inc.

Use of the "keyboard" Apple logo (Option-Shift-K) for commercial purposes without the prior written consent of Apple may constitute trademark infringement and unfair competition in violation of federal and state laws.

No licenses, express or implied, are granted with respect to any of the technology described in this document. Apple retains all intellectual property rights associated with the technology described in this document. This document is intended to assist application developers to develop applications only for Apple-labeled or Apple-licensed computers.

Every effort has been made to ensure that the information in this document is accurate. Apple is not responsible for typographical errors.

Apple Computer, Inc.

1 Infinite Loop

Cupertino, CA 95014

408-996-1010

Apple, the Apple logo, Mac, Mac OS, and QuickTime are trademarks of Apple Computer, Inc., registered in the United States and other countries.

Times is a trademark of Heidelberger Druckmaschinen AG, available from Linotype Library GmbH.

Simultaneously published in the United States and Canada.

Even though Apple has reviewed this manual, APPLE MAKES NO WARRANTY OR REPRESENTATION, EITHER EXPRESS OR IMPLIED, WITH RESPECT TO THIS MANUAL, ITS QUALITY, ACCURACY, MERCHANTABILITY, OR FITNESS FOR A PARTICULAR PURPOSE. AS A RESULT, THIS MANUAL IS SOLD "AS IS," AND YOU, THE PURCHASER, ARE ASSUMING THE ENTIRE RISK AS TO ITS QUALITY AND ACCURACY. IN NO EVENT WILL APPLE BE LIABLE FOR DIRECT, INDIRECT, SPECIAL, INCIDENTAL, OR CONSEQUENTIAL DAMAGES RESULTING FROM ANY DEFECT OR INACCURACY IN THIS MANUAL, even if advised of the possibility of such damages. THE WARRANTY AND REMEDIES SET FORTH ABOVE ARE EXCLUSIVE AND IN LIEU OF ALL OTHERS, ORAL OR WRITTEN, EXPRESS OR IMPLIED. No Apple dealer, agent, or employee is authorized to make any modification, extension, or addition to this warranty.

Some states do not allow the exclusion or limitation of implied warranties or liability for incidental or consequential damages, so the above limitation or exclusion may not apply to you. This warranty gives you specific legal rights, and you may also have other rights which **vary from state to state.**

Apple Certificate Library Functional Specification

January 13, 2005

1.0 Scope

This document describes the functions of the Apple Certificate Library (“CL”) for the Mac OS X platform.

2.0 Functional Specification

This section is targeted towards developers who will use the CL. It focuses on the API provided by the CL, mainly in terms of the certificate fields (components) which this library supports.

In general, the CL performs four functions:

- Assemble X.509 certificates from components expressed in standard CDSA data types, and sign and encode those certificates.
- Verify existing certificates given a public key or another certificate.
- Decode and parse existing X.509 certificates, allowing an app to access (read) individual fields.
- Decode and parse existing X.509 Certificate Revocation Lists (CRLs), allowing an app to access (read) individual fields.

Currently Unsupported Functions:

- Per the CDSA spec, the CL operates only on certificates in memory. It knows nothing of persistent storage, LDAP servers, or web-resident certificate authorities. Operations requiring the use of these resources is performed elsewhere.
- The current version of the CL does not create or sign Certificate Revocation Lists (CRLs).
- The CL does not manipulate Certificate Bundles; it only operates on single certificates.

2.1 Supported Certificate Fields

Refer to the CDSA specification, May 2000, chapters 10 and 31 for background on this section.

The CDSA spec provides a number of way to access individual certificate components (or “fields” per the CDSA spec). Access to specific fields, when either parsing or constructing a certificate, is via OID/Value pairs expressed as CSSM_FIELDS. CSSM_FIELD.FieldOid is an OID which identifies a particular certificate field. CSSM_FIELD.FieldValue contains, in one of many formats (see below), the actual certificate component.

In hopes of maintaining a straightforward API, the current design generally provides access to certificate fields as C structures – not as BER-encoded blobs, LDAP strings, etc. When decoding and parsing a certificate, these C structures are allocated by the CL on the app’s behalf; a pointer to a given C structure is returned in a CSSM_FIELD.FieldValue.Data pointer. The CL will free a CSSM_FIELD and/or all of its referents via CL_FreeFields() or CL_FreeFieldValue().

The following tables list the accessible certificate fields, the OID associated with each field, and the C structure by which the field is represented (“rep” in the table) when passed between the app and the CL.

2.1.1 Standard X.509 V3 Certificate fields

The C structures for these fields are defined in <cdsa/x509defs.h>. The OIDs are defined in <cdsa/oidscert.h>.

Version

OID: CSSMOID_X509V1Version

Rep: BER-encoded integer, MS byte first, length in FieldValue.Length

Note: This field is optional; if not present, the default value of 0 (indicating version 1) should be inferred by the app.

Serial Number

OID: CSSMOID_X509V1SerialNumber

Rep: BER-encoded integer, MS byte first, length in FieldValue.Length

Algorithm Identifier in To-be-signed certificate

OID: CSSMOID_X509V1SignatureAlgorithmTBS

Rep: CSSMOID_X509_ALGORITHM_IDENTIFIER

Algorithm Identifier in certificate

OID: CSSMOID_X509V1SignatureAlgorithm

Rep: CSSMOID_X509_ALGORITHM_IDENTIFIER

Note: This field is read-only; it can not be set during a CertCreateTemplate operation.

Issuer

OID: CSSMOID_X509V1IssuerNameCStruct
Rep: C struct : CSSM_X509_NAME

Subject

OID: CSSMOID_X509V1SubjectNameCStruct
Rep: C struct : CSSM_X509_NAME

Issuer, normalized

OID: CSSMOID_X509V1IssuerName
Rep: raw bytes containing the DER encoding of the normalized issuer name. Normalization consists of converting all text to upper case, removing leading and trailing whitespace, and removing all redundant whitespace.

Subject, normalized

OID: CSSMOID_X509V1SubjectName
Rep: raw bytes containing the DER encoding of the normalized subject name..

Issuer, encoded

OID: CSSMOID_X509V1IssuerNameStd
Rep: raw bytes containing the DER encoding of the issuer name.

Subject, encoded

OID: CSSMOID_X509V1SubjectNameStd
Rep: raw bytes containing the DER encoding of the issuer name.

Validity not before

OID: CSSMOID_X509V1ValidityNotBefore
Rep: C struct : CSSM_X509_TIME

Validity not after

OID: CSSMOID_X509V1ValidityNotAfter
Rep: C struct : CSSM_X509_TIME

Issuer Unique ID

OID: CSSMOID_X509V1CertificateIssuerUniqueId
Rep: raw bytes (already DER-decoded, length in FieldValue.Length)
Note: This field is optional and no default value exists.

Subject Unique ID

OID: CSSMOID_X509V1CertificateSubjectUniqueId
Rep: raw bytes (already DER-decoded, length in FieldValue.Length)
Note: This field is optional and no default value exists.

Subject Public Key Info

OID: CSSMOID_X509V1SubjectPublicKeyCStruct

Rep: C struct : CSSM_X509_SUBJECT_PUBLIC_KEY_INFO

Note: When creating a template, this field is mutually exclusive with “Subject Public Key” (below). If it is necessary to specify algorithm parameters, use this version.

Subject Public Key

OID: CSSMOID_CSSMKeyStruct

Rep: CSSM_KEY struct

Note: When creating a template, this field is mutually exclusive with “Subject Public Key Info” (above).

Issuer, Normalized and Encoded

OID: CSSMOID_X509V1IssuerName

Rep: DER-encoded normalized issuer name. This field is intended to be used when comparing certificates’ subject and issuer names. Per RFC 3280, 4.1.2.4, when comparing subject and issuer names, case is ignored and leading, trailing, and multiple whitespace characters are ignored.

Note: This field is read-only; it can not be set during a CertCreateTemplate operation.

Subject, Normalized and Encoded

OID: CSSMOID_X509V1SubjectName

Rep: DER-encoded normalized subject name. See description of previous field for more info.

Note: This field is read-only; it can not be set during a CertCreateTemplate operation.

Signature

OID: CSSMOID_X509V1Signature

Rep: Raw signature bytes.

Note: This field is read-only; it can not be set during a CertCreateTemplate operation.

2.1.2 Standard X.509 V3 CRL fields

All of these fields are currently read-only. The CL cannot create CRLs.

Fully parsed CRL struct

OID: CSSMOID_X509V2CRLSignedCrlCStruct

Rep: CSSM_X509_SIGNED_CRL

Version

OID: CSSMOID_X509V2CRLVersion

Rep: BER-encoded integer, MS byte first, length in FieldValue.Length

Note: This field is optional; if not present, the default value of 0 (indicating version 1) should be inferred by the app.

Issuer

OID: CSSMOID_X509V1IssuerNameCStruct
Rep: C struct : CSSM_X509_NAME

Issuer, normalized

OID: CSSMOID_X509V1IssuerName
Rep: raw bytes containing the DER encoding of the normalized issuer name. Normalization consists of converting all text to upper case, removing leading and trailing whitespace, and removing all redundant whitespace.

ThisUpdate

OID: CSSMOID_X509V1CRLThisUpdate
Rep: C struct : CSSM_X509_TIME

NextUpdate

OID: CSSMOID_X509V1CRLNextUpdate
Rep: C struct : CSSM_X509_TIME

Algorithm Identifier in To-be-signed CRL

OID: CSSMOID_X509V1SignatureAlgorithmTBS
Rep: CSSMOID_X509_ALGORITHM_IDENTIFIER

2.1.3 Extensions

<Note: Please refer to RFC 3280, section 4, for information on Certificate extensions. You'll need to thoroughly understand the RFC in order to effectively use extensions.>

The CL can decode and parse certificate many of the extensions defined in RFC 3280. The CDSA spec does not provide OIDs or C structs for accessing these extensions. Thus the C structures for supported extensions are defined in the Apple-specific header <Security/certextensions.h>. OIDs for supported extensions are in <Security/oidscert.h>. Other extensions may be encountered beyond what the CL can interpret; the general scheme is that extensions which are **not** understood by the CL have a value of `CSSMOID_X509V3CertificateExtensionCStruct` in `CSSM_FIELD.FieldOid`; extensions which **are** understood by the CL are reported as fields with the appropriate extension-specific OID in `CSSM_FIELD.FieldOid`. Note that for a given certificate or CRL, multiple extensions with OID `X509V3CertificateExtensionCStruct` can exist.

All extensions are expressed in `CSSM_FIELD.FieldValue.Data` as a pointer to a `CSSM_X509_EXTENSION`. In the case where the CL understands (and has decoded and parsed) the extension, the `CSSM_X509_EXTENSION` struct is defined as follows:

<code>ExtnId</code>	The OID associated with the extension
<code>Critical</code>	As per the encoded extension
<code>Format</code>	<code>CSSM_X509_DATAFORMAT_PARSED</code>
<code>Value.parsedValue</code>	Pointer to extension-specific C struct from <code>certextensions.h</code>
<code>BERValue</code>	The <code>BER_Encoded</code> extension (“ <code>extnValue</code> ” in X.509/RFC3280 terminology)

In the general case of an extension which is NOT understood by the CL, the `CSSM_X509_EXTENSION` struct is defined as follows:

<code>ExtnId</code>	The OID associated with the extension
<code>Critical</code>	As per the encoded extension
<code>Format</code>	<code>CSSM_X509_FORMAT_ENCODED</code>
<code>Value</code>	<code>NULL</code>
<code>BERValue</code>	The <code>BER_Encoded</code> extension (“ <code>extnValue</code> ” in X.509/RFC3280 terminology)

Note that in both cases, the raw DER-encoded extension is always available in `CSSM_X509_EXTENSION.BERValue`.

2.1.3 Extensions common to Certificates and CRLs

Unparsed/undecoded extension

OID: CSSMOID_X509V3CertificateExtensionCStruct

Rep: CSSM_X509_EXTENSION, with NULL value.parsedValue, valid BERValue

Note: In this case, the FieldOid value (X509V3CertificateExtensionCStruct) is **not** the same as the value in CSSM_X509_EXTENSION.ExtnId. The latter is the OID from the cert; the former is an OID specific to the CL. For extensions which **are** understood and parsed by the CL, these two fields are identical.

Authority Key ID

OID: CSSMOID_AuthorityKeyIdentifier

Rep: CSSM_X509_EXTENSION , with Value.parsedValue pointing to a CE_AuthorityKeyID

Subject alternate name

OID: CSSMOID_SubjectAltName

Rep: CSSM_X509_EXTENSION , with Value.parsedValue pointing to a CE_GeneralNames

Issuer alternate name

OID: CSSMOID_IssuerAltName

Rep: CSSM_X509_EXTENSION , with Value.parsedValue pointing to a CE_GeneralNames

2.1.3 Certificate Extensions

Key Usage

OID: CSSMOID_KeyUsage

Rep: CSSM_X509_EXTENSION , with Value.parsedValue pointing to a CE_KeyUsage

Basic constraints

OID: CSSMOID_BasicConstraints

Rep: CSSM_X509_EXTENSION , with Value.parsedValue pointing to a CE_BasicConstraints

Extended key usage

OID: CSSMOID_ExtendedKeyUsage

Rep: CSSM_X509_EXTENSION , with Value.parsedValue pointing to a CE_ExtendedKeyUsage

Subject key ID

OID: CSSMOID_SubjectKeyIdentifier

Rep: CSSM_X509_EXTENSION , with Value.parsedValue pointing to a CE_SubjectKeyID

Cert policies

OID: CSSMOID_CertificatePolicies

Rep: CSSM_X509_EXTENSION , with Value.parsedValue pointing to a CE_CertPolicies

Netscape cert type

OID: CSSMOID_NetscapeCertType

Rep: CSSM_X509_EXTENSION , with Value.parsedValue pointing to a CE_NetscapeCertType

CRL Distribution Points

OID: CSSMOID_CrIDistributionPoints

Rep: CSSM_X509_EXTENSION , with Value.parsedValue pointing to a CE_CRLDistPointsSyntax

Authority Info Access

OID: CSSMOID_AuthorityInfoAccess

Rep: CSSM_X509_EXTENSION , with Value.parsedValue pointing to a CE_AuthorityInfoAccess

Subject Info Access

OID: CSSMOID_SubjectInfoAccess

Rep: CSSM_X509_EXTENSION , with Value.parsedValue pointing to a CE_AuthorityInfoAccess

2.1.4 CRL Extensions

CRL Number

OID: CSSMOID_CrINumber

Rep: CSSM_X509_EXTENSION , with Value.parsedValue pointing to a CE_CrINumber (a uint32)

Delta CRL

OID: CSSMOID_DeltaCrIIndicator

Rep: CSSM_X509_EXTENSION , with Value.parsedValue pointing to a CE_CrINumber (a uint32)

3.0 Revision History

<u>Revision</u>	<u>Date</u>	<u>Change</u>
0.1	8/7/2000	Initial distribution.
0.2	8/8/2000	Elaborated on Extension mechanism.
0.3	8/23/2000	Changed Usage note for CSSMOID_X509V1Version Modified Signature algorithm OIDs and usage notes Fixed typo in Validity Not Before field Changed CertExtensions.h to certextensions.h
0.4	9/14/2000	Flagged some extensions as read-only. Noted mutual exclusivity of some extensions. Clarified OID usage for non-understood extensions.
0.5	10/25/2000	Added normalized & encoded subject/issuer name fields. Fixed usage notes for fields pertaining to Subject Public Key.
1.0	1/13/2005	Update for Tiger. Added CRL info. Added numerous fields and extensions.