Document Number: DSP1096

Date: 2011-09-16

Version: 1.0.0

Certificate Management Profile

Document Type: Specification

Document Status: DMTF Standard

Document Language: en-US

Contents

Figures

Tables

Foreword

The *Certificate Management Profile* (DSP1096) was prepared by the Security Working Group of DMTF.

DMTF is a not-for-profit association of industry members dedicated to promoting enterprise and systems management and interoperability. For information about the DMTF, see http://www.dmtf.org.

Acknowledgments

The DMTF acknowledges the following individuals for their contributions to this document:

- Khachatur Papanyan – Dell
- George Ericson – EMC
- Jeff Hilland – HP
- David Hines – Intel
- Hemal Shah – Broadcom
- Sharon L. Smith – Intel

Introduction

The information in this specification is intended to be sufficient for a provider or consumer of this data to identify unambiguously the classes, properties, methods, and values that are mandatory to be instantiated and manipulated to represent and manage users and groups that are modeled using the DMTF Common Information Model (CIM) core and extended model definitions.

The target audience for this specification is implementers who are writing CIM-based providers or consumers of management interfaces that represent the component described in this document.

Document conventions

Typographical conventions

The following typographical conventions are used in this document:

- Document titles are marked in *italics*.
- Important terms that are used for the first time are marked in *italics*.
- ABNF rules are in `monospaced font`.

ABNF usage conventions

Format definitions in this document are specified using ABNF (see RFC5234), with the following deviations:

- Literal strings are to be interpreted as case-sensitive Unicode characters, as opposed to the definition in RFC5234 that interprets literal strings as case-insensitive US-ASCII characters.

Certificate Management Profile

1 Scope

The *Certificate Management Profile* specializes the *Credential Management Profile* and extends the management capability of the referencing profiles by adding the capability to model and manage X509 certificates of the public key infrastructure (PKI). This profile is not intended to serve as a mechanism for the digital identification. Creation, storage, and management of X509 certificates, public private key pairs, certificate revocation lists (CRL) and certificate signing requests (CSR) is detailed. Profile registration for the schema implementation version information is also described.

2 Normative references

The following referenced documents are indispensable for the application of this document. For dated or versioned references, only the edition cited (including any corrigenda or DMTF update versions) applies. For references without a date or version, the latest published edition of the referenced document (including any corrigenda or DMTF update versions) applies.

DMTF DSP0200, *CIM Operations over HTTP 1.3,*
http://www.dmtf.org/standards/published_documents/DSP0200_1.3.pdf

DMTF DSP0004, *CIM Infrastructure Specification 2.6,*
http://www.dmtf.org/standards/published_documents/DSP0004_2.6.pdf

DMTF DSP1001, *Management Profile Specification Usage Guide 1.0,*
http://www.dmtf.org/standards/published_documents/DSP1001_1.0.pdf

DMTF DSP1033, *Profile Registration Profile 1.0,*
http://www.dmtf.org/standards/published_documents/DSP1033_1.0.pdf

DMTF DSP1082, *Credential Management Profile 1.0*,
http://www.dmtf.org/standards/published_documents/DSP1082_1.0.pdf

IETF RFC 2986, *PKCS #10: Certification Request Syntax Specification, Version 1.7*, November 2000,
http://www.ietf.org/rfc/rfc2986.txt

IETF RFC3280, *Internet X.509 Public Key Infrastructure: Certificate and Certificate Revocation List (CRL) Profile*, April 2002, http://www.ietf.org/rfc/rfc3280.txt

IETF RFC4514, *Lightweight Directory Access Protocol (LDAP): String Representation of Distinguished Names*, June 2006, http://www.ietf.org/rfc/rfc4514.txt

IETF RFC5234, *Augmented BNF for Syntax Specifications: ABNF*, January 2008,
http://www.ietf.org/rfc/rfc5234.txt

ISO/IEC Directives, Part 2, *Rules for the structure and drafting of International Standards*,
http://isotc.iso.org/livelink/livelink.exe?func=ll&objId=4230456&objAction=browse&sort=subtype

3 Terms and definitions

In this document, some terms have a specific meaning beyond the normal English meaning. Those terms are defined in this clause.

The terms "shall" ("required"), "shall not," "should" ("recommended"), "should not" ("not recommended"), "may," "need not" ("not required"), "can" and "cannot" in this document are to be interpreted as described in ISO/IEC Directives, Part 2, Annex H. The terms in parenthesis are alternatives for the preceding term, for use in exceptional cases when the preceding term cannot be used for linguistic reasons. Note that ISO/IEC Directives, Part 2, Annex H specifies additional alternatives. Occurrences of such additional alternatives shall be interpreted in their normal English meaning.

The terms "clause," "subclause," "paragraph," and "annex" in this document are to be interpreted as described in ISO/IEC Directives, Part 2, Clause 5.

The terms "normative" and "informative" in this document are to be interpreted as described in ISO/IEC Directives, Part 2, Clause 3. In this document, clauses, subclauses, or annexes labeled "(informative)" do not contain normative content. Notes and examples are always informative elements.

The terms defined in DSP0004, DSP0200, and DSP1001 apply to this document. The following additional terms are used in this document.

3.1
Asymmetric Key Pair
public/private key pair represented by CIM_UnsignedCredential

3.2
Container Keystore
the single instance of CIM_Keystore that contains a given PKI Credential Instance
The instance of CIM_Keystore is associated to the PKI Credential Instance through the CIM_MemberOfCollection association.

3.3
Managing Service
The instance of CIM_CertificateManagementService that is managing a given PKI Credential Instance
The instance of CIM_CertificateManagementService is associated to the Container Keystore of the given PKI Credential Instance through the CIM_ServiceAffectsElement association.

3.4
PKI Credential Instance
an instance of CIM_UnsignedCredential, CIM_X509Certificate, or CIM_X509CRL representing a PKI credential

4 Symbols and abbreviated terms

The abbreviations defined in DSP0004, DSP0200, and DSP1001 apply to this document. The following additional abbreviations are used in this document.

3.5
CRL
Certification Revocation List

3.6
CSR
Certificate Signing Request

3.7
PKI
Public Key Infrastructure

5 Synopsis

Profile Name: Certificate Management

Version: 1.0.0

Organization: DMTF

CIM schema version: 2.29

Central Class: CIM_CertificateManagementService

Scoping Class: CIM_System

The *Certificate Management Profile* specializes the *Credential Management Profile* to extend the management capability of the referencing profiles by adding the capability to represent and manage X509 certificates.

The Central Class of the *Certificate Management Profile* shall be CIM_CertificateManagementService. The Central Instance shall be an instance of CIM_CertificateManagementService. The Scoping Class shall be CIM_System. The Scoping Instance shall be the instance of CIM_System that is associated with the Central Instance through the CIM_HostedService association.

Table 1 lists the profiles related to the *Certificate Management Profile*.

Table 1 – Referenced profiles

Profile Name	Organization	Version	Relationship	Behavior
Credential Management	DMTF	1.0	Specializes	
Profile Registration	DMTF	1.0	Mandatory	

6 Description

The *Certificate Management Profile* describes the properties and methods for X509 certificate management in a managed system. This profile does not provide a mechanism for an application to authenticate using certificates but rather to manage the certificates that are used by the managed system for the authentication.

Figure 1 represents the class schema for the profile. For simplicity, the prefix *CIM_* has been removed from the names of the classes.

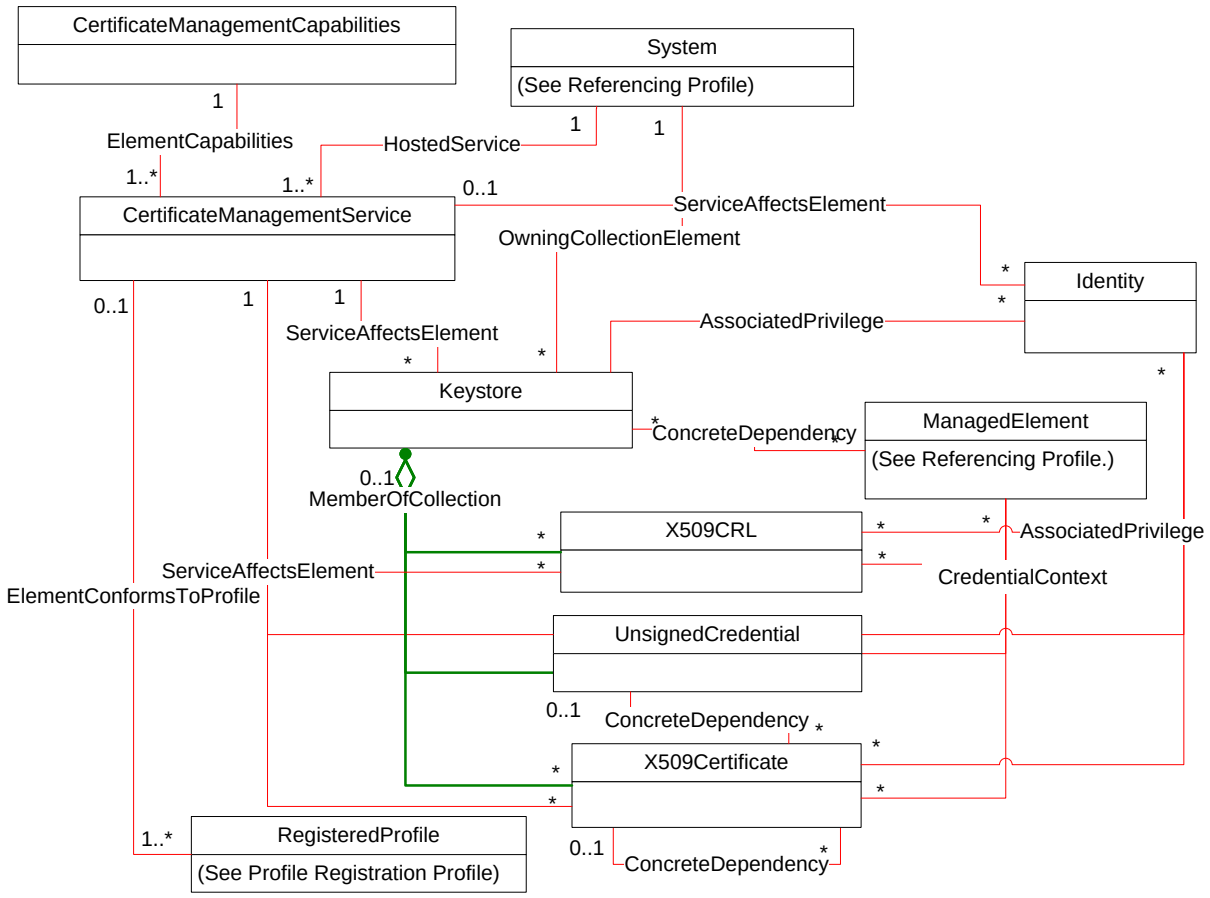


Figure 1 – Certificate Management Profile: Class diagram

The *Certificate Management Profile* consists of the representation of X509 certificate management services using CIM_CertificateManagementService, key stores accumulating PKI-based credentials using CIM_Keystore, and the PKI based credentials using CIM_UnsignedCredential, CIM_X509CRL, and CIM_X509Certificate.

6.1 Key store

Key store aggregates the PKI-based credentials used by the managed system's services. There are two usage scenarios predominant with key stores:

1) Another entity connects with the managed system service and presents its digital identification for the managed system's service to verify. To verify the connected entity's identification, the managed system service looks up a key store for a matching trusted certificate to verify against. In this case, the key store is identified as a trusted key store.

2) The managed system's service connects to an entity where the connected entity requires the managed system's service to present its digital identification. The managed system service's digital identification is stored in the key store identified as an owned key store.

For example:

1) A web service running on the managed system utilizes the owned key store for the X509 certificate that is presented to the connecting web client.

2) LDAP client services on the managed system utilize the trusted key store to store trusted X509 certificates to verify against the certificate presented by the LDAP server, which the managed system connects to in the process of authentication.

Both the owned key store and the trusted key store are represented by the CIM_Keystore class which is associated to the CIM_ManagedElement class representing the service that utilizes the credential store by the CIM_ConcreteDependency association.

6.2 Public key infrastructure

Public key infrastructure is used by the managed system for establishing trust. The credentials used in public key infrastructure are represented by the classes derived from the CIM_Credential class, such as CIM_UnsignedCredential, CIM_X509Certificate, and CIM_X509CRL.

6.2.1 Asymmetric key pair

Asymmetric key pair on a managed system is represented by CIM_UnsignedCredential. The asymmetric key pair can be used as an input for generating a Certificate Signing Request or a self-signed certificate through the methods in the CIM_CertificateManagementService. CIM_CertificateManagementService also contains an extrinsic method for importing an asymmetric key pair.

6.2.2 X.509 certificate

X.509 certificate is represented by the CIM_X509Certificate class. CIM_X509Certificate contains properties corresponding to the X.509 certificate fields defined in RFC3280.

If the X.509 certificate was generated using imported asymmetric key, then the CIM_UnsignedCredential instance representing the asymmetric key is associated to the CIM_X509Certifacte instance representing the generated X.509 certificate through CIM_ConcreteDependency association, where the Antecedent property references the CIM_UnsignedCredential instance and the Dependent property references the CIM_X509Certificate instance.

If the managed element that uses X.509 certificate for authentication is represented by a subclass of CIM_ManagedElement, this instance of the subclass of CIM_ManagedElement will be associated with the CIM_X509Certificate instance using the CIM_CredentialContext association or will be associated with the CIM_Keystore instance aggregating the CIM_X509Certificate instance using the CIM_ConcreteDependency association. Thus, CIM_ConcreteDependency and CIM_CredentialContext represent the usage of the credentials.

Certificate chains are also represented by the CIM_ConcreteDependency class associating the CIM_X509Certificate instances representing the signer and signed certificates. CIM_CertificateManagementService contains methods for creating, importing, and exporting certificates.

6.2.3 X.509 certificate revocation list (CRL)

X.509 CRL is represented by CIM_X509CRL class. CIM_X509CRL contains properties corresponding to the X.509 CRL fields defined in RFC3280. CIM_CertificateManagementService contains a method for applying the encoded X.509 CRL received from the Certificate Authority onto a specific key store.

6.3 Authorization

X.509 certificates and keys may have different levels of access authorization. An authorized entity is represented by a security principal through the CIM_Identity class. A security principal may be authorized to access the credential store or a particular credential within a key store. The AssociatedPrivilege

association contains the privileges of the security principal as well as references to the key store, the certificate/key, or both.

When an implementation has the ability to authorize on both levels (per key store and per the certificate/key), the implementation calculates the effective authorization privileges for a particular security principal by combining the key stores and its members' privileges in one of the following ways:

1) Collection Privileges Override – The effective privileges are the key store privileges overriding a particular certificate/key's privileges.
2) Member Privileges Override – The effective credential privileges are the particular certificate/key privileges overriding the key store's privileges.
3) Collection-Member Privileges Union – The effective credential privilege is the union of the key store privileges and the particular certificate/key privileges.
4) Collection-Member Privileges Intersection – The effective credential privilege is the intersection of the key store privileges and the particular certificate/key privileges.

The implementation supporting the key store level and member level privileges will implement one of the above methodologies for calculating the effective privilege for a key/certificate. The CertificateManagementCapabilities class will advertise which of these methodologies the implementation supports.

7 Implementation

This clause details the requirements related to the arrangement of instances and their properties for implementations of this profile.

7.1 Key store

The requirements for the instances of CIM_Keystore and PKI Credential Instances are defined in the *Credential Management Profile*.

7.2 Certificate management service

This subclause details the requirements for the certificate management service and the representation of its capabilities.

7.2.1 CIM_CertificateManagementService

Each PKI-based credential represented by a PKI Credential Instance shall be managed by one and only one certificate management service represented by an instance of CIM_CertificateManagementService, also referred to as a Managing Service (see 3.3). Each Container Keystore shall be associated to an instance of CIM_CertificateManagementService. The PKI Credential Instances within the Container Credential Store shall be managed by the CIM_CertificateManagementService instance, which the Container Keystore is associated through CIM_ServiceAffectsElement.

7.2.2 CIM_CertificateManagementCapabilities

There shall be one and only one instance of CIM_CertificateMangementCapabilities associated with an instance of CIM_CertificateManagementService representing the capabilities of a certificate management service.

If the AsymmetricKeyGeneration property is TRUE, then the KeyAlgorithmSupportedProperty shall have a non empty, non-NULL value.

If the SupportedMethods array property contains a value of 103 (ImportEncodedCertificates) or 106 (ApplyCRL), then the InputFormatsSupported array property shall have a non empty, non-NULL value.

If the SupportedMethods array property contains a value of 105 (ExportEncodedCertificate), then the OutputFormatsSupported array property shall have a non empty, non-NULL value.

7.3 Certificate chain

Certificate chains may be represented. If the complete chain of certificates is implemented, the requirements in this subclause shall apply.

Each certificate in the certificate chain shall be represented by an instance of CIM_X509Certificate. The instance of CIM_X509Certificate representing the signing certificate shall be associated with instances of CIM_X509Certificate that represent the signed certificates using CIM_ConcreteDependency, where the Antecedent property shall reference the instance of CIM_X509Certificate representing the signing certificate and the Dependent property shall reference the instance of CIM_X509Certificate representing the signed certificate.

7.4 Authorization

This subclause details the requirements that are in addition to the authorization requirements in the *Credential Management Profile*.

If the security principal authorization is implemented, then the CIM_Identity instance representing the security principal shall be associated with a CIM_KeyStore instance or PKI Credential Instance through the CIM_AssociatedPrivilege association.

For the CIM_Keystore instance referenced by CIM_AssociatedPrivilege, the referenced CIM_Identity shall be authorized to perform the intrinsic operations or execute extrinsic methods in the “Intrinsic Operations” and “Extrinsic Methods” columns of Table 2, if and only if the CIM_AssociatedPrivilege.Activities property contains the value from the “CIM_AssociatedPrivilege.Activities” column of the respective row of Table 2.

Table 2 – CIM_AssociatedPrivilege.Activities mapping to PKI Credential Instance operations

CIM_AssociatedPrivilege. Activities	Credential Operation	Intrinsic Operations	Extrinsic Methods
2 (Create) or 6 (Write)	Import a key/certificate into a key store	Modify Instance on PKI Credential Instance	ImportPublicPrivateKeyPair( ) CreateCertificateSigningRequest( ) CreateSelfSignedCertificate( ) ImportEncodedCertificates( ) ImportCertificates( ) ApplyCRL( ) ApplyDecodedCRL( )
5 (Read)	Export a key/certificate from a key store	Get on PKI Credential Instance Enumerate on PKI Credential Instance	ExportEncodedCertificates( )
3 (Delete)	Delete a key/certificate from a key store	Delete on PKI Credential Instance	

For the PKI Credential Instance referenced by CIM_AssociatedPrivilege, the referenced CIM_Identity shall be authorized to perform the intrinsic operations or execute extrinsic methods in the “Intrinsic Operations” and “Extrinsic Methods” columns of Table 3, if and only if the CIM_AssociatedPrivilege.Activities property contains the value from “CIM_AssociatedPrivilege.Activities” column of the respective row of Table 3.

Table 3 – CIM_AssociatedPrivilege.Activities mapping to PKI Credential Instance operations

CIM_AssociatedPrivilege. Activities	Credential Operation	Intrinsic Operations	Extrinsic Methods
6 (Write)	Modify a key or certificate	Modify Instance on PKI Credential Instance	
5 (Read)	Get a key or certificate	Get on PKI Credential Instance Enumerate on PKI Credential Instance	ExportEncodedCertificates( )
3 (Delete)	Delete a key or certificate	Delete on PKI Credential Instance	

8 Methods

This clause details the requirements for supporting intrinsic operations and extrinsic methods for the CIM elements defined by this profile.

8.1 CIM_CertificateManagementService.ImportPublicPrivateKeyPair()

The ImportPublicPrivateKeyPair( ) method provides the ability to import a public/private key pair to be used by the managed system.

The ImportPublicPrivateKeyPair( ) method’s return code values shall be as specified in Table 4, where the method execution behavior matches the return code description. The ImportPublicPrivateKeyPair( ) method’s parameters are specified in

Table 5.

No standard messages are defined for this method.

Table 4 – CIM_CertificateManagementService.ImportPublicPrivateKeyPair( ) method: Return code values

Value	Description
0	Operation completed successfully
1	Operation unsupported
2	Failed
4096	Job created – only Job output parameter shall be returned.

454

455

456 **Table 5 – CIM_CertificateManagementService.ImportPublicPrivateKeyPair( ) method: Parameters**

Qualifiers	Name	Type	Description/Values
IN, REQ, OctetString	PublicKey[]	string	Public key as an octet string array of one element. The public key shall be contained in the first element of the array. All other array element shall be ignored.
IN, REQ, OctetString	PrivateKey[]	string	Private key as an octet string array of one element. The private key shall be contained in the first element of the array. All other array element shall be ignored.
IN, REQ	Keystore	CIM_Keystore REF	References the key store to which the asymmetric key is to be added
IN	CredentialContext	CIM_ManagedElement REF	References an instance of CIM_ManagedElement that will utilize the imported credential
IN	Usage[]	uint16	Describes how the managed element, referenced by the CredentialContext parameter, utilizes the imported credential
OUT	Job	CIM_ConcreteJob REF	Reference to the ConcreteJob created to track the execution initiated by the method invocation.
OUT, REQ	PPKPCredential	CIM_UnsignedCredential REF	References the instance representing the imported public/private key pair.

457 ### 8.1.1 CIM_CertificateManagementService.ImportPublicPrivateKeyPair( ) conditional
458 support

459 If the SupportedMethods property array of the associated instance of
460 CIM_CertificateManagementCapabilities contains the value 2 (ImportPublicPrivateKeyPair), the
461 ImportPublicPrivateKeyPair( ) method shall be implemented and shall not return the value 1 (Not
462 Supported).

463 If the SupportedMethods property array of the associated instance of
464 CIM_CertificateManagementCapabilities does not contain the value 2 (ImportPublicPrivateKeyPair), the
465 ImportPublicPrivateKeyPair( ) method shall not be implemented or shall always return the value 1 (Not
466 Supported).

467 ### 8.1.2 Parameter validation

468 If the Usage parameter has a non-NULL value, then the CredentialContext parameter shall have a non-
469 NULL value. If the Usage parameter has non-NULL value and the CredentialContext parameter is NULL,
470 then the method shall return 2 (Failed).

8.1.3 Required privileges

The method invocation shall fail if the Activities property of the CIM_AssociatedPrivilege instance, referencing the CIM_Identity instance representing the executor's security principal, does not contain the value 2 (Create) or 6 (Write).

8.2 CIM_CertificateManagementService.CreateKeystore( )

The CreateKeystore( ) method provides the ability to create new key stores.

The CreateKeystore( ) method's return code values shall be as specified in Table 6 where the method execution behavior matches the return code description. The CreateKeystore( ) method's parameters are specified in Table 7.

No standard messages are defined for this method.

Table 6 – CIM_CertificateManagementService.CreateKeystore( ) method: Return code values

Value	Description
0	Operation completed successfully
1	Operation unsupported
2	Failed
4096	Job created – only Job output parameter shall be returned.

Table 7 – CIM_CertificateManagementService.CreateKeystore( ) method: Parameters

Qualifiers	Name	Type	Description/Values
IN, REQ	Keystore	String	Embedded instance representing the new key store
IN, REQ	OwningSystem	CIM_System REF	Reference to the owning managed system for the key store
IN	KeystoreUtilizers[]	CIM_ManagedElement REF	Array of references to the CIM_ManagedElement subclass instances that will utilize the new key store
OUT	Job	CIM_ConcreteJob REF	Reference to the ConcreteJob created to track the execution initiated by the method invocation
OUT, REQ	Keystore	CIM_Keystore REF	Reference to the newly created CIM_Keystore instance representing the new key store

8.2.1 CIM_CertificateManagementService.CreateKeystore( ) conditional support

If the SupportedMethods property array of the associated instance of CIM_CertificateManagementCapabilities contains the value 3 (CreateKeystore), the CreateKeystore( ) method shall be implemented and shall not return the value 1 (Not Supported).

If the SupportedMethods property array of the associated instance of CIM_CertificateManagementCapabilities does not contain the value 3 (CreateKeystore), the CreateKeystore( ) method shall not be implemented or shall always return the value 1 (Not Supported).

8.2.2 Parameter validation

The Keystore parameter shall have non-NULL values for the Usage property of the CIM_Keystore embedded instance. If the Usage property of the CIM_Keystore embedded instance of the Keystore parameter is NULL or is missing, the method shall return a value of 2 (Failed).

8.3 CIM_CertificateManagementService.CreateCertificateSigningRequest( )

The CreateCertificateSigningRequest( ) method provides the ability to create CSRs based on PKCS#10 referenced in RFC2986.

The CreateCertificateSigningRequest( ) method's return code values shall be as specified in Table 8, where the method execution behavior matches the return code description. The CreateCertificateSigningRequest( ) method's parameters are specified in Table 9.

No standard messages are defined for this method.

Table 8 – CIM_CertificateManagementService.CreateCertificateSigningRequest( ) method: Return code values

Value	Description
0	Operation completed successfully
1	Operation unsupported
2	Failed
4096	Job created – only Job output parameter shall be returned.

Table 9 – CIM_CertificateManagementService.CreateCertificateSigningRequest( ) method: Parameters

Qualifiers	Name	Type	Description/Values
IN	Subject	string	Subject shall contain information as required by section 4.1.2.4 of RFC3280, formatted based on RFC4514.
IN	AltSubject	string	String containing alternate subject identifier
IN	PublicKeyAlgorithm	uint16	The algorithm used for generating the public/private key pair. This parameter shall be supported if the AsymmetricKeyGeneration property of the associated CIM_CertificateManagementCapabilities instance is TRUE.
IN	PublicKeySize	uint16	The length of the public key. This parameter shall be supported if the AsymmetricKeyGeneration property of the associated CIM_CertificateManagementCapabilities is TRUE and if the PublicPrivateKeyPair is not specified.
IN	PublicPrivateKeyPair	CIM_UnsignedCredential REF	Reference to the instance of CIM_UnsignedCredential representing the public/private key pair

Qualifiers	Name	Type	Description/Values
IN	ExtendedKeyUsageValue[]	string	Indicates one or more purposes for which the certified public key may be used. The value at each index describes the key usage value based on the type at the corresponding index of ExtendedKeyUsageType[] parameter array.
IN	ExtendedKeyUsageType[]	uint16	Indicates the type for ExtendedKeyUsageValue based on the ASN.1 GeneralName types. The value at each index describes the type for ExtendedKeyUsageValue parameter at the corresponding index.
IN	SignatureAlgorithm	uint16	Indicates the signature algorithm used to sign the CSR
IN, REQ	OutputFormat	uint16	Indicates the output format of the CSR
OUT	Job	CIM_ConcreteJob REF	Reference to the ConcreteJob created to track the execution initiated by the method invocation.
OUT, REQ, OctetString	CSR[]	string	CSR as a single-element octet string array. The entire CSR shall be contained in the first array element. All other array elements shall be ignored, if present.

8.3.1 CIM_CertificateManagementService.CreateCertificateSigningRequest( ) conditional support

If the SupportedMethods property array of the associated instance of CIM_CertificateManagementCapabilities contains the value 101 (CreateCertificateSigningRequest), the CreateCertificateSigningRequest( ) method shall be implemented and shall not return the value 1 (Not Supported).

If the SupportedMethods property array of the associated instance of CIM_CertificateManagementCapabilities does not contain the value 101 (CreateCertificateSigningRequest), the CreateCertificateSigningRequest( ) method shall not be implemented or shall always return the value 1 (Not Supported).

8.3.2 Parameter validation

If the PublicPrivateKeyPair parameter is NULL and the AsymmetricKeyGeneration property of the associated instance of CIM_CertificateManagementCapabilities has the value FALSE, then the method shall return a value of 2 (Failed).

If the PublicPrivateKeyPair parameter is NULL and the PublicKeyAlgorithm or PublicKeySize parameters are NULL, then the method shall return a value of 2 (Failed).

If the PublicKeyAlgorithm parameter is not NULL and the KeyAlgorithmSupported array property of the associated instance of CIM_CertificateManagementCapabilities does not contain the value of the PublicKeyAlgorithm parameter, then the method shall return a value of 2 (Failed).

If the Subject and AltSubject parameters are both NULL, then the method shall return a value of 2 (Failed).

If the OutputFormat parameter does not contain a value specified in the SupportedOutputFormats property array of the associated CIM_CertificateManagementCapabilities instance, then the method shall return a value 2 (Failed).

If the CIM_CertificateManagementCapabilities.SupportedSignatureAlgorithms property is implemented and the SignatureAlgorithm parameter does not contain a value specified in the SupportedSignatureAlgorithms property array, then the method shall return a value 2 (Failed).

8.3.3 Required privileges

The method invocation shall fail if the Activities property of the CIM_AssociatedPrivilege instance, referencing the CIM_Identity instance representing the executor's security principal, does not contain a value of 2 (Create) or 6 (Write).

8.4 CIM_CertificateManagementService.CreateSelfSignedCertificate( )

The CreateSelfSignedCertificate( ) method provides the ability to create an X509 self-signed certificate either based on the pre-generated public-private key pair or based on the method execution generated public private key-pair.

The CreateSelfSignedCertificate( ) method's return code values shall be as specified in Table 10 where the method execution behavior matches the return code description. The CreateSelfSignedCertificate( ) method's parameters are specified in Table 11.

If the pre-generated public-private key pair is used by providing a valid PublicPrivateKeyPair parameter, then the generated X.509 certificate in the SelfSignedCertificate output parameter shall reference the CIM_UnsignedCredential instance representing the PublicPrivateKeyPair through CIM_ConreteDependency association, where the Antecedent property references the CIM_UnsignedCredential instance and the Dependent property references the CIM_X509Certificate instance.

No standard messages are defined for this method.

Table 10 – CIM_CertificateManagementService.CreateSelfSignedCertificate( ) method: Return code values

Value	Description
0	Operation completed successfully
1	Operation unsupported
2	Failed
4096	Job created – only Job output parameter shall be returned.

Table 11 – CIM_CertificateManagementService.CreateSelfSignedCertificate( ) method: Parameters

Qualifiers	Name	Type	Description/Values
IN	Subject	string	Subject shall contain information as required by section 4.1.2.4 of RFC3280, formatted based on RFC4514
IN	AltSubject	string	String containing alternate subject identifier

Qualifiers	Name	Type	Description/Values
IN	PublicKeyAlgorithm	uint16	The algorithm used for generating public/private key pair. This parameter shall be supported if the AsymmetricKeyGeneration property of the associated CIM_CertificateManagementCapabilities is TRUE and if the PublicPrivateKeyPair is not specified.
IN	PublicKeySize	uint16	The length of the public key. This parameter shall be supported if the AsymmetricKeyGeneration property of the associated CIM_CertificateManagementCapabilities is TRUE.
IN	PublicPrivateKeyPair	CIM_UnsignedCredential REF	Reference to the instance of CIM_UnsignedCredential representing the public/private key pair
IN	Keystore	CIM_Keystore REF	References the key store to add the newly created X509 self-signed certificate
IN	CredentialContext	CIM_ManagedElement REF	References an instance of CIM_ManagedElement that will utilize the newly created X509 self-signed certificate
IN	Usage	uint16	Describes how the managed element, referenced by CredentialContext parameter, utilizes the self-signed certificate
IN	SignatureAlgorithm	uint16	Indicates the signature algorithm used to sign the X509 self-signed certificate
OUT	Job	CIM_ConcreteJob REF	Reference to the ConcreteJob instance created to track the execution initiated by the method invocation
OUT, REQ	SelfSignedCertificate	CIM_X509Certificate REF	References the newly created X509 self-signed certificate

8.4.1 CIM_CertificateManagementService.CreateSelfSignedCertificate( ) conditional support

If the SupportedMethods property array of the associated instance of CIM_CertificateManagementCapabilities contains the value 102 (CreateSelfSignedCertificate), the CreateSelfSignedCertificate( ) method shall be implemented and shall not return the value 1 (Not Supported).

If the SupportedMethods property array of the associated instance of CIM_CertificateManagementCapabilities does not contain the value 102 (CreateSelfSignedCertificate), the CreateSelfSignedCertificate( ) method shall not be implemented or shall always return the value 1 (Not Supported).

8.4.2 Parameter validation

If the PublicPrivateKeyPair parameter is NULL and the AsymmetricKeyGeneration property of the associated instance of CIM_CertificateManagementCapabilities has the value FALSE, then the method shall return a value of 2 (Failed).

If the PublicPrivateKeyPair parameter is NULL and the PublicKeyAlgorithm or PublicKeySize parameters are NULL, then the method shall return a value of 2 (Failed).

If the PublicKeyAlgorithm parameter is not NULL and the KeyAlgorithmSupported array property of the associated instance of CIM_CertificateManagementCapabilities does not contain the value of the PublicKeyAlgorithm parameter, then the method shall return a value of 2 (Failed).

If the Subject and AltSubject parameters are both NULL, the method shall return a value of 2 (Failed).

If the CIM_CertificateManagementCapabilities.SupportedSignatureAlgorithms property is implemented and the SignatureAlgorithm parameter does not contain a value specified in the SupportedSignatureAlgorithms property array, then the method shall return a value 2 (Failed).

8.4.3 Required privileges

The method invocation shall fail if the Activities property of the CIM_AssociatedPrivilege, referencing the CIM_Identity instance representing the executor's security principal, does not contain a value of 2 (Create) or 6 (Write).

8.5 CIM_CertificateManagementService.ImportEncodedCertificates( )

The ImportEncodedCertificates( ) method provides the ability to import encoded X509 certificates.

The ImportEncodedCertificates( ) method's return code values shall be as specified in Table 12, where the method execution behavior matches the return code description. The ImportEncodedCertificates( ) method's parameters are specified in Table 13.

No standard messages are defined for this method.

Table 12 – CIM_CertificateManagementService.ImportEncodedCertificates( ) method: Return code values

Value	Description
0	Operation completed successfully
1	Operation unsupported
2	Failed
4096	Job created – only Job output parameter shall be returned.

Table 13 – CIM_CertificateManagementService.ImportEncodedCertificates( ) method: Parameters

Qualifiers	Name	Type	Description/Values
IN, REQ	EncodedCertificates[]	string	Single-element octet string array representing the encoded X509 certificates
IN, REQ	Format	uint16	Specifies the format for the encoding of the encoded X509 certificates

Qualifiers	Name	Type	Description/Values
IN	Keystore	CIM_Keystore REF	References the key store to which the newly imported X509 certificates are to be added
IN	CredentialContext	CIM_ManagedElement REF	References an instance of CIM_ManagedElement that will utilize the newly imported X509 certificates
IN	Usage[]	uint16	Describes how the managed element, referenced by CredentialContext parameter, utilizes the imported certificates
OUT	Job	CIM_ConcreteJob REF	Reference to the ConcreteJob created to track the execution initiated by the method invocation
OUT, REQ	NewCertificates[]	CIM_X509Certificate REF	References newly imported X509 certificates

8.5.1 CIM_CertificateManagementService.ImportEncodedCertificates( ) conditional support

If the SupportedMethods property array of the associated instance of CIM_CertificateManagementCapabilities contains the value 103 (ImportEncodedCertificates), the ImportEncodedCertificates( ) method shall be implemented and shall not return the value 1 (Not Supported).

If the SupportedMethods property array of the associated instance of CIM_CertificateManagementCapabilities does not contain the value 103 (ImportEncodedCertificates), the ImportEncodedCertificates( ) method shall not be implemented or shall always return the value 1 (Not Supported).

8.5.2 Parameter validation

If the Format parameter does not contain a value from the InputFormatsSupported array property of the associated instance of CIM_CertificateManagementCapabilities, then the method shall return a value of 2 (Failed).

8.5.3 Required privileges

The method invocation shall fail if the Activities property of the CIM_AssociatedPrivilege, referencing the CIM_Identity instance representing the executor’s security principal, does not contain 2 (Create) or 6 (Write) values.

8.6 CIM_CertificateManagementService.ImportCertificates( )

The ImportCertificates( ) method provides the ability to import X509 certificates as embedded instances of the CIM_X509Certificate class.

The ImportCertificates( ) method’s return code values shall be as specified in Table 14, where the method execution behavior matches the return code description. The ImportCertificates( ) method’s parameters are specified in Table 15.

No standard messages are defined for this method.

615 **Table 14 – CIM_CertificateManagementService.ImportCertificates( ) method: Return code values**

Value	Description
0	Operation completed successfully
1	Operation unsupported
2	Failed
4096	Job created – only Job output parameter shall be returned.

616 **Table 15 – CIM_CertificateManagementService.ImportCertificates( ) method: Parameters**

Qualifiers	Name	Type	Description/Values
IN, REQ	InputCertificates[]	string	Embedded instances of CIM_X509Certificate representing the imported certificates
IN	Keystore	CIM_Keystore REF	References the key store to which to add the newly imported X509 certificates
IN	CredentialContext	CIM_ManagedElement REF	References an instance of CIM_ManagedElement that will utilize the newly imported X509 certificates
IN	Usage[]	uint16	Describes how the managed element, referenced by the CredentialContext parameter, utilizes the imported certificates
OUT	Job	CIM_ConcreteJob REF	Reference to the ConcreteJob instance created to track the execution initiated by the method invocation
OUT, REQ	NewCertificates[]	CIM_X509Certificate REF	References newly imported X509 certificates

617 ### 8.6.1 CIM_CertificateManagementService.ImportCertificates( ) conditional support

618 If the SupportedMethods property array of the associated instance of
619 CIM_CertificateManagementCapabilities contains the value 104 (ImportCertificates), the
620 ImportCertificates( ) method shall be implemented and shall not return the value 1 (Not Supported).

621 If the SupportedMethods property array of the associated instance of
622 CIM_CertificateManagementCapabilities does not contain the value 104 (ImportCertificates), the
623 ImportCertificates( ) method shall not be implemented or shall always return the value 1 (Not Supported).

624 ### 8.6.2 Required privileges

625 The method invocation shall fail if the Activities property of the CIM_AssociatedPrivilege, referencing the
626 CIM_Identity instance representing the executor's security principal, does not contain a value of 2
627 (Create) or 6 (Write).

628 ## 8.7 CIM_CertificateManagementService.ExportEncodedCertificates( )

629 The ExportEncodedCertificates( ) method provides the ability to export X509 certificates using a specified
630 encoded format.

The ExportEncodedCertificates( ) method's return code values shall be as specified in Table 16 where the method execution behavior matches the return code description. The ExportEncodedCertificates( ) method's parameters are specified in Table 17.

No standard messages are defined for this method.

Table 16 – CIM_CertificateManagementService.ExportEncodedCertificates( ) method: Return code values

Value	Description
0	Operation completed successfully
1	Operation unsupported
2	Failed
4096	Job created – only Job output parameter shall be returned.

Table 17 – CIM_CertificateManagementService.ExportEncodedCertificates( ) method: Parameters

Qualifiers	Name	Type	Description/Values
IN, REQ	CertificatesToExport[]	CIM_X509Certificate REF	Reference to the CIM_ManagedElement for which metrics will be controlled
IN, REQ	Format	uint16	Specifies the format for the encoding of the encoded X509 certificates to export
OUT	Job	CIM_ConcreteJob REF	Reference to the ConcreteJob created to track the execution initiated by the method invocation
OUT, REQ, OctetString	EncodedCertificates[]	string	Single-element octet string array representing the exported X509 certificates. All the X509 certificates shall be contained in the first element of the array. All other array element shall be ignored.

8.7.1 CIM_CertificateManagementService.ExportEncodedCertificates( ) conditional support

If the SupportedMethods property array of the associated instance of CIM_CertificateManagementCapabilities contains the value 105 (ExportEncodedCertificate), the ExportEncodedCertificates( ) method shall be implemented and shall not return the value 1 (Not Supported).

If the SupportedMethods property array of the associated instance of CIM_CertificateManagementCapabilities does not contain the value 105 (ExportEncodedCertificate), the ExportEncodedCertificates( ) method shall not be implemented or shall always return the value 1 (Not Supported).

8.7.2 Parameter validation

If the Format parameter does not contain a value from the OutputFormatsSupported array property of the associated instance of CIM_CertificateManagementCapabilities, then the method shall return a value of 2 (Failed).

If the instances of CIM_X509Certificate in the adjacent indexes of the CertificatesToExport array parameter are not associated with each other using the CIM_ConcreteDependency association, where

654 the instance at the lower index is referenced by the Antecedent property and the instance at the higher
655 index is referenced by the Dependent property, the method shall return a value of 2 (Failed).

656 8.7.3 Required privileges

657 The method invocation shall fail if the Activities property of the CIM_AssociatedPrivilege instance,
658 referencing the CIM_Identity instance representing the executor's security principal, does not contain a
659 value of 5 (Read).

660 8.8 CIM_CertificateManagementService.ApplyCRL()

661 The ApplyCRL( ) method provides the ability to apply CRL in an encoded format.

662 The ApplyCRL( ) method's return code values shall be as specified in Table 18, where the method
663 execution behavior matches the return code description. The ApplyCRL( ) method's parameters are
664 specified in Table 19.

665 No standard messages are defined for this method.

666 **Table 18 – CIM_CertificateManagementService.ApplyCRL( ) method: Return code values**

Value	Description
0	Operation completed successfully
1	Operation unsupported
2	Failed
4096	Job created – only Job output parameter shall be returned.

667 **Table 19 – CIM_CertificateManagementService.ApplyCRL( ) method: Parameters**

Qualifiers	Name	Type	Description/Values
IN, REQ, OctetString	EncodedCRL[]	string	Single-element string octet array representing the encoded CRL. The encoded CRL shall be contained in the first element of the array. All other array element shall be ignored.
IN, REQ	Format	uint16	Specifies the format for the encoding of the encoded CRL
IN	Keystore	CIM_Keystore REF	References the key store to which the CRL will be applied
IN	CredentialContext	CIM_ManagedElement REF	References an instance of CIM_ManagedElement for which the certificates are revoked
OUT	Job	CIM_ConcreteJob REF	Reference to the ConcreteJob created to track the execution initiated by the method invocation
OUT, REQ	AppliedCRL	CIM_X509CRL REF	References the newly applied CRL

8.8.1 CIM_CertificateManagementService.ApplyCRL( ) conditional support

If the SupportedMethods property array of the associated instance of CIM_CertificateManagementCapabilities contains the value 106 (ApplyCRL), the ApplyCRL( ) method shall be implemented and shall not return the value 1 (Not Supported).

If the SupportedMethods property array of the associated instance of CIM_CertificateManagementCapabilities does not contain the value 106 (ApplyCRL), the ApplyCRL( ) method shall not be implemented or shall always return the value 1 (Not Supported).

8.8.2 Parameter validation

If the InputFormatsSupported array property of the associated instance of CIM_CertificateManagementCapabilities does not contain the value of the Format parameter, the method shall return a value of 2 (Failed).

8.8.3 Required privileges

The method invocation shall fail if the Activities property of the CIM_AssociatedPrivilege, referencing the CIM_Identity instance representing the executor's security principal, does not contain a value of 2 (Create) or 6 (Write).

8.9 CIM_CertificateManagementService.ApplyDecodedCRL( )

The ApplyDecodedCRL( ) method provides the ability to apply CRL in the decoded format.

The ApplyDecodedCRL( ) method's return code values shall be as specified in Table 18, where the method execution behavior matches the return code description. The ApplyDecodedCRL( ) method's parameters are specified in Table 19.

No standard messages are defined for this method.

Table 20 – CIM_CertificateManagementService.ApplyDecodedCRL( ) method: Return code values

Value	Description
0	Operation completed successfully
1	Operation unsupported
2	Failed
4096	Job created – only Job output parameter shall be returned.

Table 21 – CIM_CertificateManagementService.ApplyDecodedCRL( ) method: Parameters

Qualifiers	Name	Type	Description/Values
IN, REQ	Issuer	string	Issuer shall contain information as required by section 4.1.2.4 of RFC3280, formatted based on RFC4514.
IN, REQ	SerialNumbers[]	String	Serial numbers of the certificates contained in the CRL
IN	Keystore	CIM_Keystore REF	References the key store to which the CRL will be applied

Qualifiers	Name	Type	Description/Values
IN	CredentialContext	CIM_ManagedElement REF	References an instance of CIM_ManagedElement for which the certificates are revoked
OUT	Job	CIM_ConcreteJob REF	Reference to the ConcreteJob created to track the execution initiated by the method invocation
OUT, REQ	AppliedCRL	CIM_X509CRL REF	References newly applied CRL

8.9.1 CIM_CertificateManagementService.ApplyDecodedCRL() conditional support

If the SupportedMethods property array of the associated instance of CIM_CertificateManagementCapabilities contains the value 107 (ApplyDecodedCRL), the ApplyDecodedCRL( ) method shall be implemented and shall not return the value 1 (Not Supported).

If the SupportedMethods property array of the associated instance of CIM_CertificateManagementCapabilities does not contain the value 107 (ApplyDecodedCRL), the ApplyDecodedCRL( ) method shall not be implemented or shall always return the value 1 (Not Supported).

8.9.2 Required privileges

The method invocation shall fail if the Activities property of the CIM_AssociatedPrivilege instance, referencing the CIM_Identity instance representing the executor's security principal, does not contain a value of 2 (Create) or 6 (Write).

8.10 Profile conventions for operations

For each profile class (including associations), the implementation requirements for operations, including those in the following default list, are specified in class-specific subclauses of this clause.

The default list of operations is as follows:

- GetInstance
- Associators
- AssociatorNames
- References
- ReferenceNames
- EnumerateInstances
- EnumerateInstanceNames

8.11 CIM_CertificateManagementCapabilities

All operations in the default list in 8.10 shall be implemented as defined in DSP0200.

NOTE: Related profiles may define additional requirements on operations for the profile class.

8.12 CIM_CertificateManagementService

All operations in the default list in 8.10 shall be implemented as defined in DSP0200.

NOTE: Related profiles may define additional requirements on operations for the profile class.

8.13 CIM_CredentialContext

Table 22 lists implementation requirements for operations. If implemented, these operations shall be implemented as defined in DSP0200. In addition, and unless otherwise stated in Table 22, all operations in the default list in 8.10 shall be implemented as defined in DSP0200.

NOTE: Related profiles may define additional requirements on operations for the profile class.

Table 22 – Operations: CIM_CredentialContext

Operation	Requirement	Messages
Associators	Unspecified	None
AssociatorNames	Unspecified	None
References	Unspecified	None
ReferenceNames	Unspecified	None

8.14 CIM_ConcreteDependency (CIM_Keystore)

Table 23 lists implementation requirements for operations. If implemented, these operations shall be implemented as defined in DSP0200. In addition, and unless otherwise stated in Table 23, all operations in the default list in 8.10 shall be implemented as defined in DSP0200.

NOTE: Related profiles may define additional requirements on operations for the profile class.

Table 23 – Operations: CIM_ConcreteDependency

Operation	Requirement	Messages
Associators	Unspecified	None
AssociatorNames	Unspecified	None
References	Unspecified	None
ReferenceNames	Unspecified	None

8.15 CIM_ConcreteDependency (CIM_X509Certificate)

Table 24 lists implementation requirements for operations. If implemented, these operations shall be implemented as defined in DSP0200. In addition, and unless otherwise stated in Table 24, all operations in the default list in 8.10 shall be implemented as defined in DSP0200.

NOTE: Related profiles may define additional requirements on operations for the profile class.

Table 24 – Operations: CIM_ConcreteDependency

Operation	Requirement	Messages
Associators	Unspecified	None
AssociatorNames	Unspecified	None
References	Unspecified	None
ReferenceNames	Unspecified	None

8.16 CIM_ElementCapabilities

Table 25 lists implementation requirements for operations. If implemented, these operations shall be implemented as defined in DSP0200. In addition, and unless otherwise stated in Table 25, all operations in the default list in 8.10 shall be implemented as defined in DSP0200.

NOTE: Related profiles may define additional requirements on operations for the profile class.

Table 25 – Operations: CIM_ElementCapabilities

Operation	Requirement	Messages
Associators	Unspecified	None
AssociatorNames	Unspecified	None
References	Unspecified	None
ReferenceNames	Unspecified	None

8.17 CIM_HostedService

Table 26 lists implementation requirements for operations. If implemented, these operations shall be implemented as defined in DSP0200. In addition, and unless otherwise stated in Table 26, all operations in the default list in 8.10 shall be implemented as defined in DSP0200.

NOTE: Related profiles may define additional requirements on operations for the profile class.

Table 26 – Operations: CIM_HostedService

Operation	Requirement	Messages
Associators	Unspecified	None
AssociatorNames	Unspecified	None
References	Unspecified	None
ReferenceNames	Unspecified	None

8.18 CIM_Keystore

All operations in the default list in 8.10 shall be implemented as defined in DSP0200.

NOTE: Related profiles may define additional requirements on operations for the profile class.

8.19 CIM_MemberOfCollection

Table 27 lists implementation requirements for operations. If implemented, these operations shall be implemented as defined in DSP0200. In addition, and unless otherwise stated in Table 27, all operations in the default list in 8.10 shall be implemented as defined in DSP0200.

NOTE: Related profiles may define additional requirements on operations for the profile class.

Table 27 – Operations: CIM_MemberOfCollection

Operation	Requirement	Messages
Associators	Unspecified	None
AssociatorNames	Unspecified	None
References	Unspecified	None
ReferenceNames	Unspecified	None

8.20 CIM_OwningCollectionElement

Table 28 lists implementation requirements for operations. If implemented, these operations shall be implemented as defined in DSP0200. In addition, and unless otherwise stated in Table 28, all operations in the default list in 8.10 shall be implemented as defined in DSP0200.

NOTE: Related profiles may define additional requirements on operations for the profile class.

Table 28 – Operations: CIM_OwningCollectionElement

Operation	Requirement	Messages
Associators	Unspecified	None
AssociatorNames	Unspecified	None
References	Unspecified	None
ReferenceNames	Unspecified	None

8.21 CIM_RegisteredProfile

All operations in the default list in 8.10 shall be implemented as defined in DSP0200.

NOTE: Related profiles may define additional requirements on operations for the profile class.

8.22 CIM_ServiceAffectsElement

Table 29 lists implementation requirements for operations. If implemented, these operations shall be implemented as defined in DSP0200. In addition, and unless otherwise stated in Table 29, all operations in the default list in 8.10 shall be implemented as defined in DSP0200.

NOTE: Related profiles may define additional requirements on operations for the profile class.

Table 29 – Operations: CIM_ServiceAffectsElement

Operation	Requirement	Messages
Associators	Unspecified	None
AssociatorNames	Unspecified	None
References	Unspecified	None
ReferenceNames	Unspecified	None

8.23 CIM_UnsignedCredential

All operations in the default list in 8.10 shall be implemented as defined in DSP0200.

NOTE: Related profiles may define additional requirements on operations for the profile class.

8.24 CIM_X509Certificate

Table 30 lists implementation requirements for operations. If implemented, these operations shall be implemented as defined in DSP0200. In addition, and unless otherwise stated in Table 30, all operations in the default list in 8.10 shall be implemented as defined in DSP0200.

NOTE: Related profiles may define additional requirements on operations for the profile class.

782 **Table 30 – Operations: CIM_X509Certificate**

Operation	Requirement	Messages
ModifyInstance	Optional	See 8.24.1.
DeleteInstance	Optional	See 8.24.2.

783 ### 8.24.1 CIM_X509Certificate — ModifyInstance

784 If the SupportedMethods property of the CIM_CertificateManagementCapabilities instance associated
785 with the Managing Service of the CIM_X509Certificate instance contains a value 107 (Modify
786 X509Certificate), then the ModifyInstance operation shall be supported.

787 If the invalidation of the certificates is supported, the IsValid property shall be modified using
788 theModifyInstance operation.

789 The ModifyInstance operation shall fail if the Activities property of the CIM_AssociatedPrivilege,
790 referencing the CIM_Identity instance representing the executor's security principal, does not contain a
791 value of 6 (Write).

792 ### 8.24.2 CIM_X509Certificate — DeleteInstance

793 If the SupportedMethods property of the CIM_CertificateManagementCapabilities instance associated
794 with the Managing Service of the CIM_X509Certificate instance contains a value of 108 (Delete
795 X509Certificate), then the DeleteInstance operation shall be supported.

796 If the removal of the certificates from the key store is supported, the DeleteInstance operation shall be
797 supported on the CIM_X509Certificate class.

798 The ModifyInstance operation shall fail if the Activities property of the CIM_AssociatedPrivilege,
799 referencing the CIM_Identity instance representing the executor's security principal, does not contain a
800 value of 3 (Delete).

801 ## 8.25 CIM_X509CRL

802 All operations in the default list in 8.10 shall be implemented as defined in DSP0200.

803 NOTE: Related profiles may define additional requirements on operations for the profile class.

804 ## 8.26 CIM_Identity

805 All operations in the default list in 8.10 shall be implemented as defined in DSP0200.

806 NOTE: Related profiles may define additional requirements on operations for the profile class.

807 ## 8.27 CIM_AssociatedPrivilege

808 Table 31 lists implementation requirements for operations. If implemented, these operations shall be
809 implemented as defined in DSP0200. In addition, and unless otherwise stated in Table 31, all operations
810 in the default list in 8.10 shall be implemented as defined in DSP0200.

811 NOTE: Related profiles may define additional requirements on operations for the profile class.

812 **Table 31 – Operations: CIM_AssociatedPrivilege**

Operation	Requirement	Messages
Associators	Unspecified	None
AssociatorNames	Unspecified	None
References	Unspecified	None
ReferenceNames	Unspecified	None

813

814 8.28 CIM_ConcreteDependency

815 Table 31 lists implementation requirements for operations. If implemented, these operations shall be
816 implemented as defined in DSP0200. In addition, and unless otherwise stated in Table 31, all operations
817 in the default list in 8.10 shall be implemented as defined in DSP0200.

818 NOTE: Related profiles may define additional requirements on operations for the profile class.

819 **Table 32 – Operations: CIM_ConcreteDependency**

Operation	Requirement	Messages
Associators	Unspecified	None
AssociatorNames	Unspecified	None
References	Unspecified	None
ReferenceNames	Unspecified	None

820

821 9 Use cases

822 This clause contains object diagrams and use cases for the *Certificate Management Profile*. The contents
823 of this clause are for informative purposes only and do not constitute normative requirements for
824 implementations of this specification.

825 9.1 Profile registration

826 Figure 2 describes one of the ways that the implementation can advertise the instantiation of the
827 *Certificate Management Profile*. Using scoping instance methodology as described in the *Profile*
828 *Registration Profile*, profile2 contains the version information for the *Certificate Management Profile*
829 implementation.

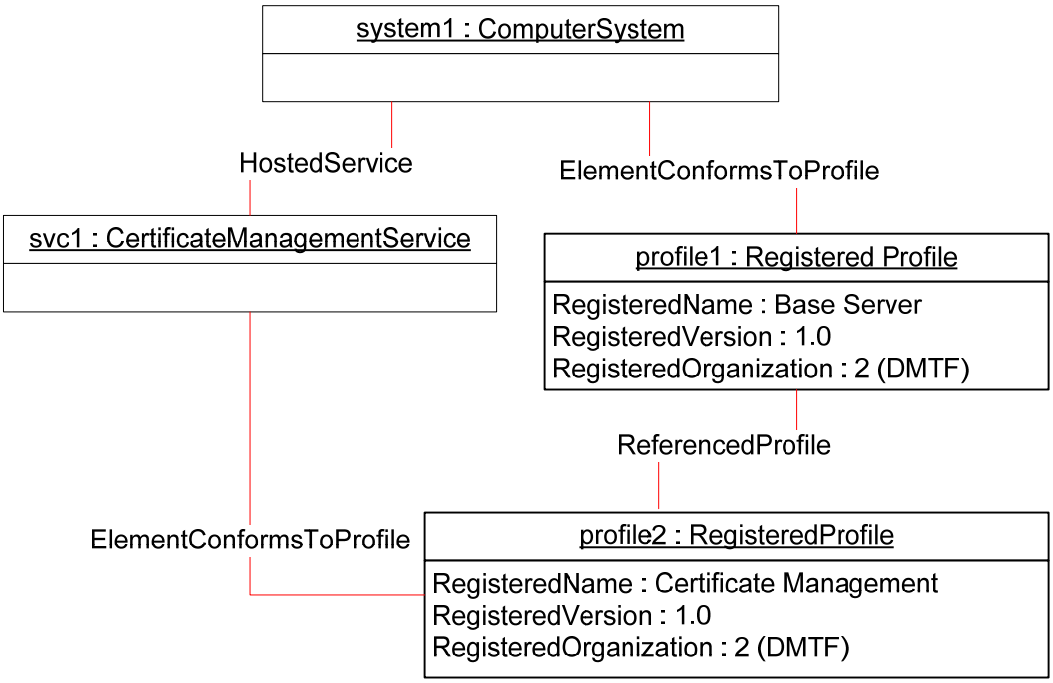


Figure 2 – Profile registration

9.2 Simple certificate management

Figure 3 represents a minimal instantiation of the *Certificate Management Profile* without key stores and certificate utilizers. Webcert1 is the only certificate managed by the service. The managed element that uses the webcert1 certificate is not instantiated, and thus the CredentialContext association does not reference the webcert1 instance. Instead, the webcert1.ElementName property describes the purpose of the certificate. Cap1 represents the capabilities of service1. Webcert1 is the only certificate that is being managed by service1. The managed element that uses the webcert1 certificate is not instantiated, and thus the CredentialContext association does not reference the webcert1 instance. Instead, the webcert1.ElementName property describes the purpose of the certificate.

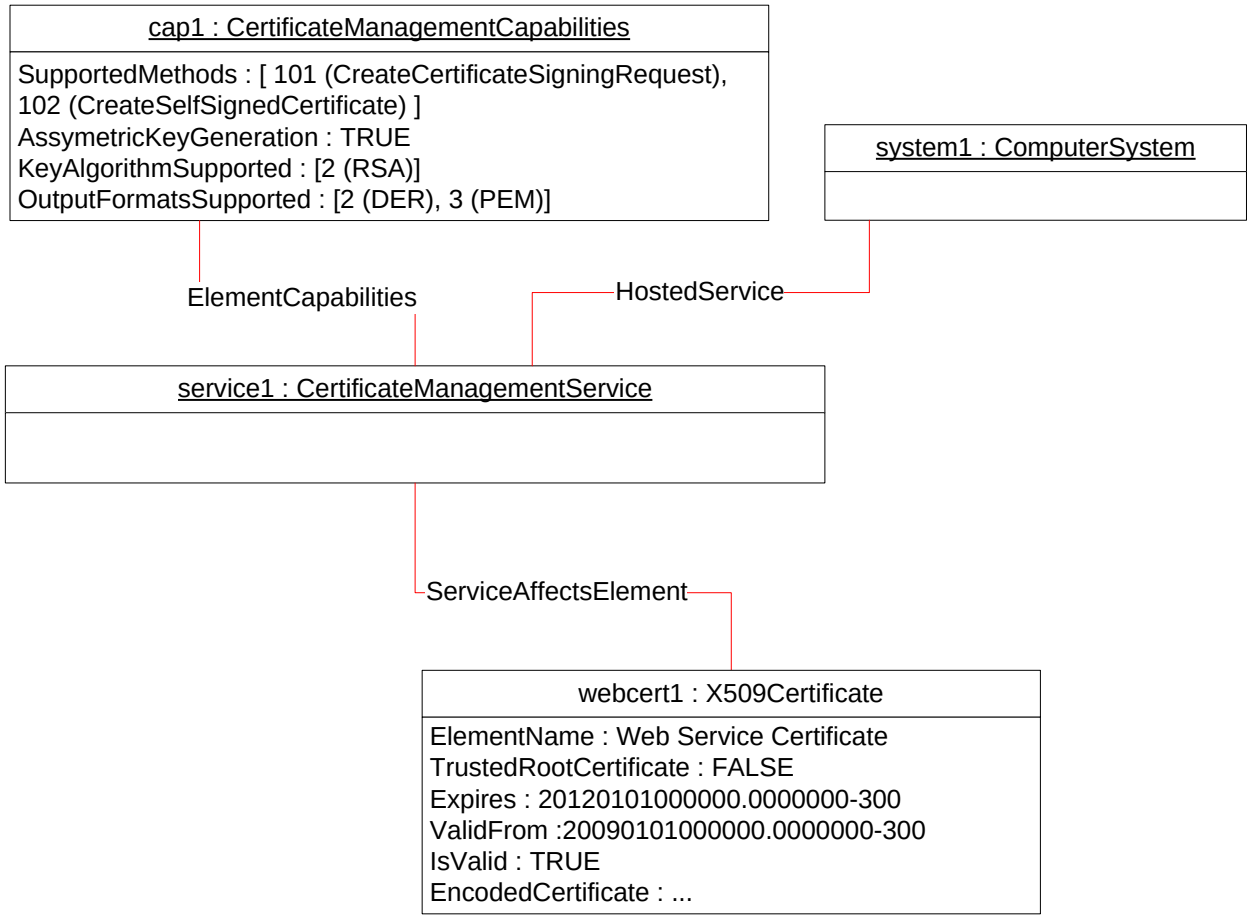
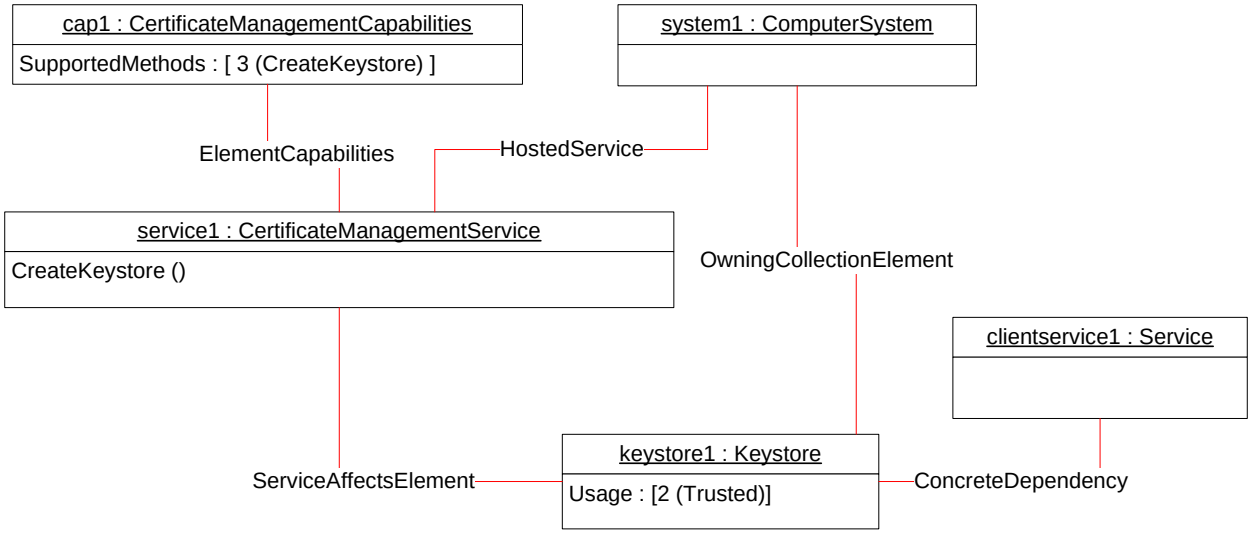


Figure 3 – Simple certificate management

9.3 Keystore

Figure 4 represents an instantiation of the *Certificate Management Profile.* In this instantiation, the instrumentation advertises the ability to create new key stores using the cap1.SupportedMethods property, which contains the value 3 (CreateKeystore). Upon successful invocation of a similarly named method on service1, a new key store will be created and will be represented by the keystore1 instance. Keystore1 properties will be equal to the Keystore embedded instance parameter's property of the invoked CreateKeystore method. The association instances referencing keystore1 are created based on the other parameters of the invoked CreateKeystore method. (See 8.2 and the method description in the CIM schema.)



852

853 **Figure 4 – Key store**

854 9.4 Import asymmetric key

855 Figure 5 represents an instantiation of the *Certificate Management Profile.* In this instantiation, the
856 instrumentation advertises the ability to import public/private key pairs using the cap1.SupportedMethods
857 property, which contains the value 2 (ImportPublicPrivateKeyPair). Upon successful invocation of a
858 similarly named method on service1, a new asymmetric key pair represented by PPKPwebservice1 will
859 be inserted in the key store represented by keystore1. The PPKPwebservice1.PublicKey property will
860 match the PublicKey parameter of the method. PPKPwebservice1 will not expose the private key for
861 security purposes. The association instances referencing PPKPwebservice1 are created based on the
862 other parameters of the invoked method. (See 8.1 and the method description in the CIM schema.)

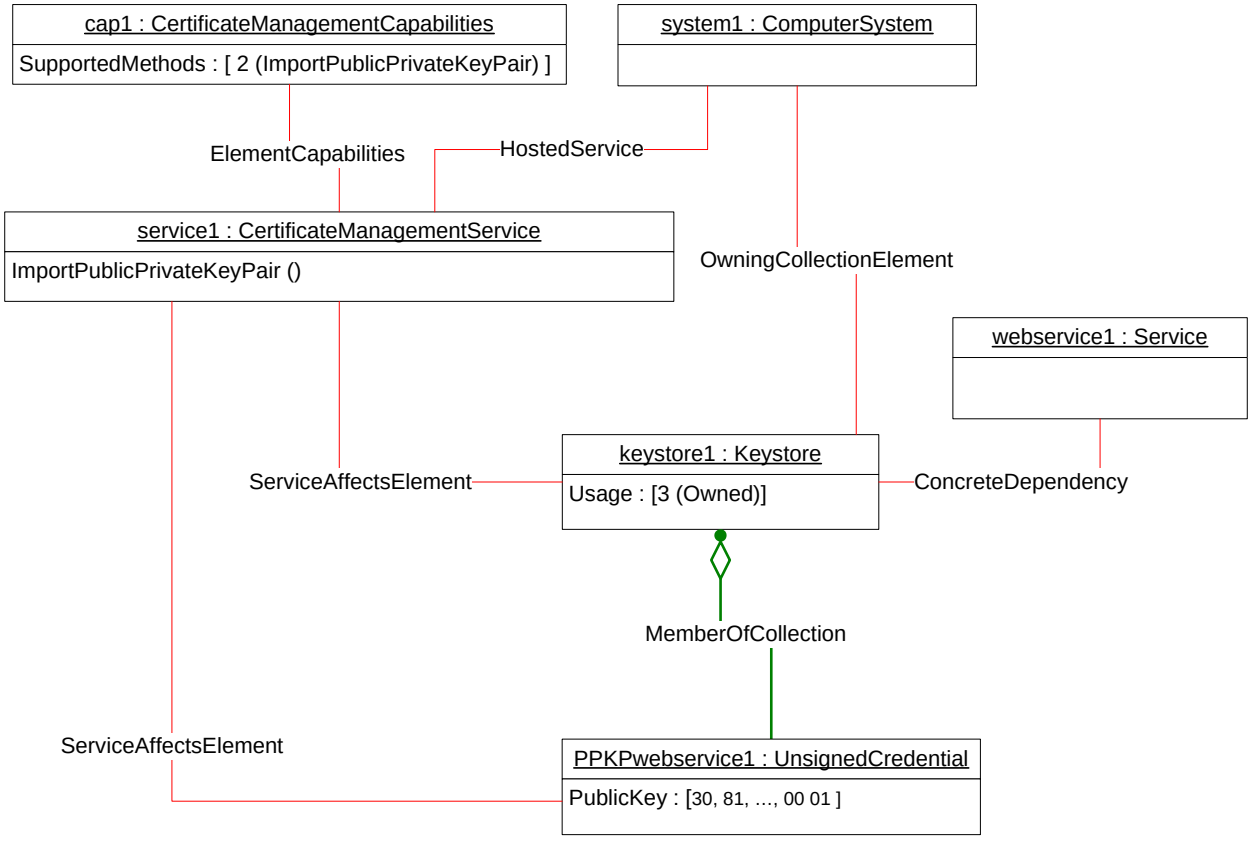


Figure 5 – Importing public/private key pair

9.5 CSR and self-signed certificate

Figure 6 represents an instantiation of the *Certificate Management Profile.* In this instantiation, the instrumentation advertises the ability to create CSRs and self-signed certificates using the cap1.SupportedMethods property, which contains the values 101 (CreateCertificateSigningRequest) and 102 (CreateSelfSignedCertificate). The cap1.AsymmetricKeyGeneration property advertises the capability of service1 to generate asymmetric keys, specifically so that if the Subject, PublicKeyAlgorithm, has PublicKeySize parameters that are valid non-NULL values, the service will be able to create either CSR or the self-signed certificate generating a public/private asymmetric key pair. KeyAlgorithmSupported advertises the value maps that could be used as the value for the PublicKeyAlgorithm parameter. If AsymmetricKeyGeneration is FALSE, to create CSR or a self-signed certificate the PublicPrivateKeyPair parameter needs to reference an instance of CIM_UnsignedCredential, such as PPKPwebservice1, that represents a previously imported public/private asymmetric key pair.

Upon successful invocation of the CreateCertificateSigningRequest method on service1, the CSR will be returned as the CSR[] output parameter. Upon successful invocation of the CreateSelfSignedCertificate method, a new instance of CIM_X509Certificate, webcert1, will be instantiated, and the reference of the instance will be returned as the SelfSignedCertificate output parameter. The association instances referencing webcert1 are created based on the other parameters of the invoked method. (See 8.3, 8.4, and the method descriptions in the CIM schema.)

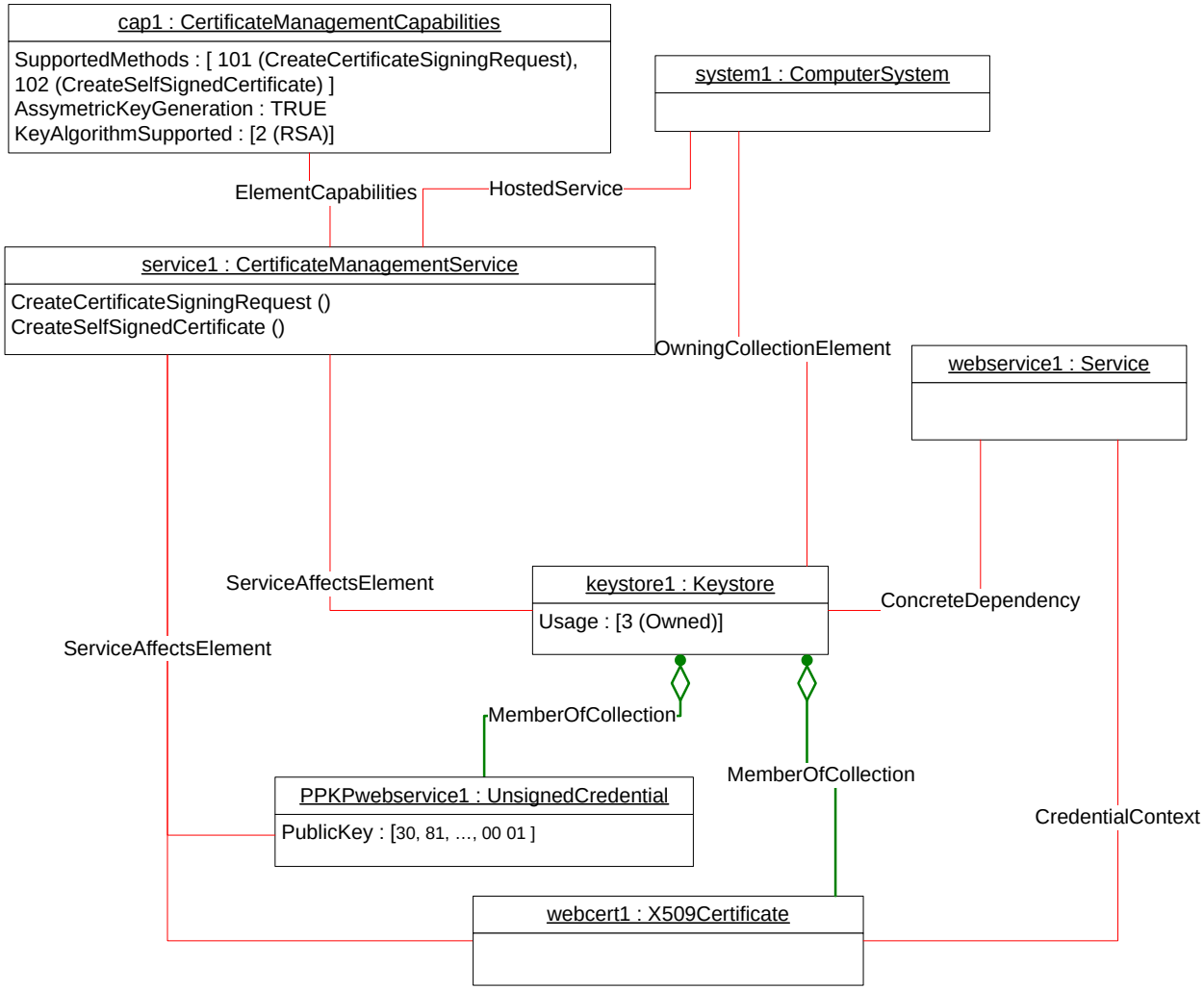


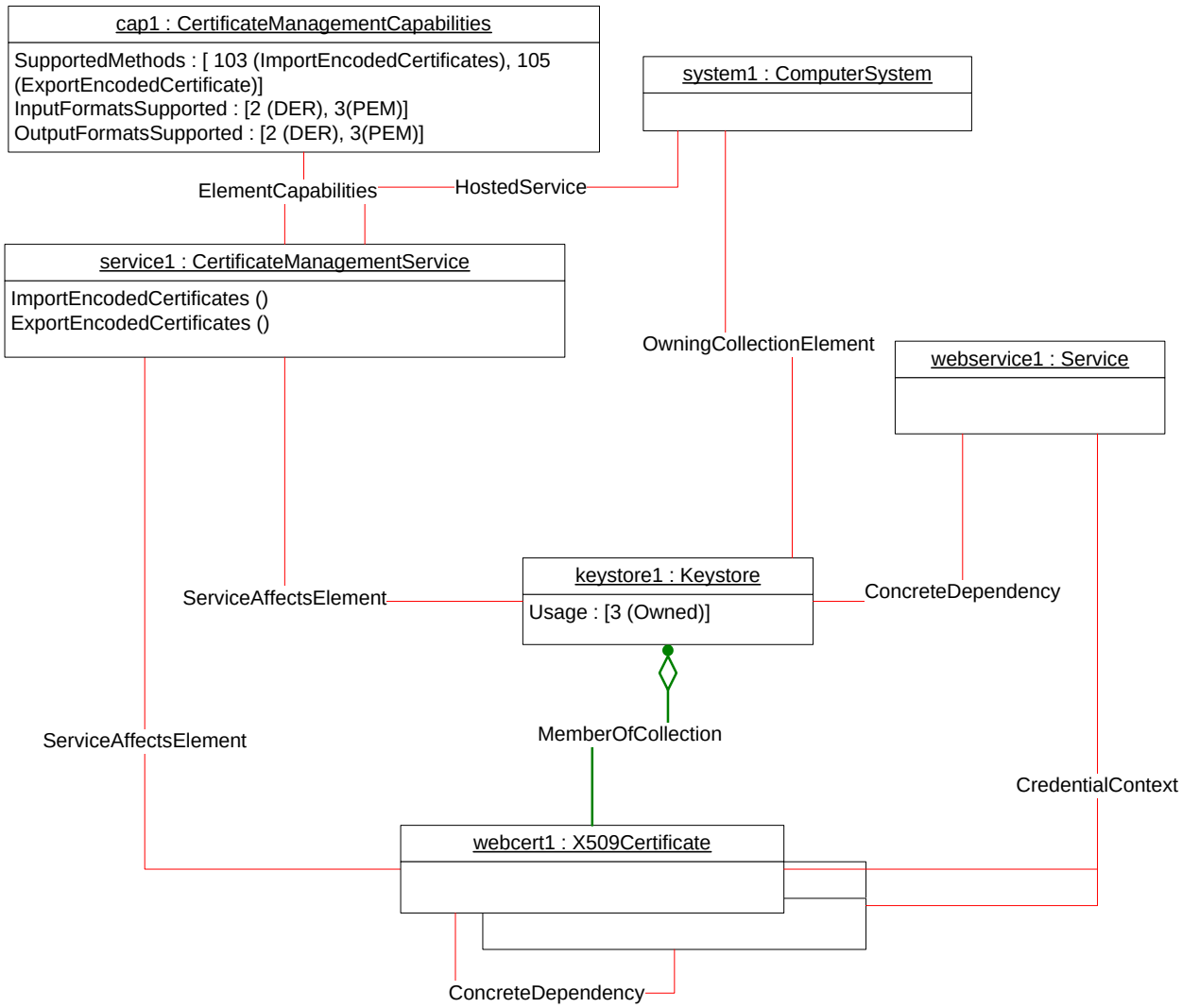
Figure 6 – Creating CSR and self-signed certificate

9.6 Import and export of certificates

Figure 7 represents an instantiation of the *Certificate Management Profile.* In this instantiation, the instrumentation advertises the ability to export and import certificates using the cap1.SupportedMethods property, which contains the values 103 (ImportEncodedCertificates) and 105 (ExportEncodedCertificate). The cap1.InputFormatsSupported property advertises the capability of service1 to import certificates in the specified formats and the value of the Format parameter on the InputEncodedCertificates method will match one of the values in this property array. Upon successful invocation of the ImportEncodedCertificates method a new instance of CIM_X509Certificate, webcert1, will be instantiated and the reference of the instance returned as the NewCertificates[] output parameter. The association instances referencing webcert1 are created based on the other parameters of the invoked method. (See 8.5 and the method descriptions in the CIM schema.)

The cap1.OutputFormatsSupported property advertises the capability of service1 to export certificates in the specified formats, and the value of the Format parameter on the ExportEncodedCertificates( ) method will match one of the values in this property array. In this case, the CertificatestoExport parameter references webcert1 and upon successful invocation of the ExportEncodedCertificates( ) method the EncodedCertificates[] array will contain the formatted version of webcert1.

901



902

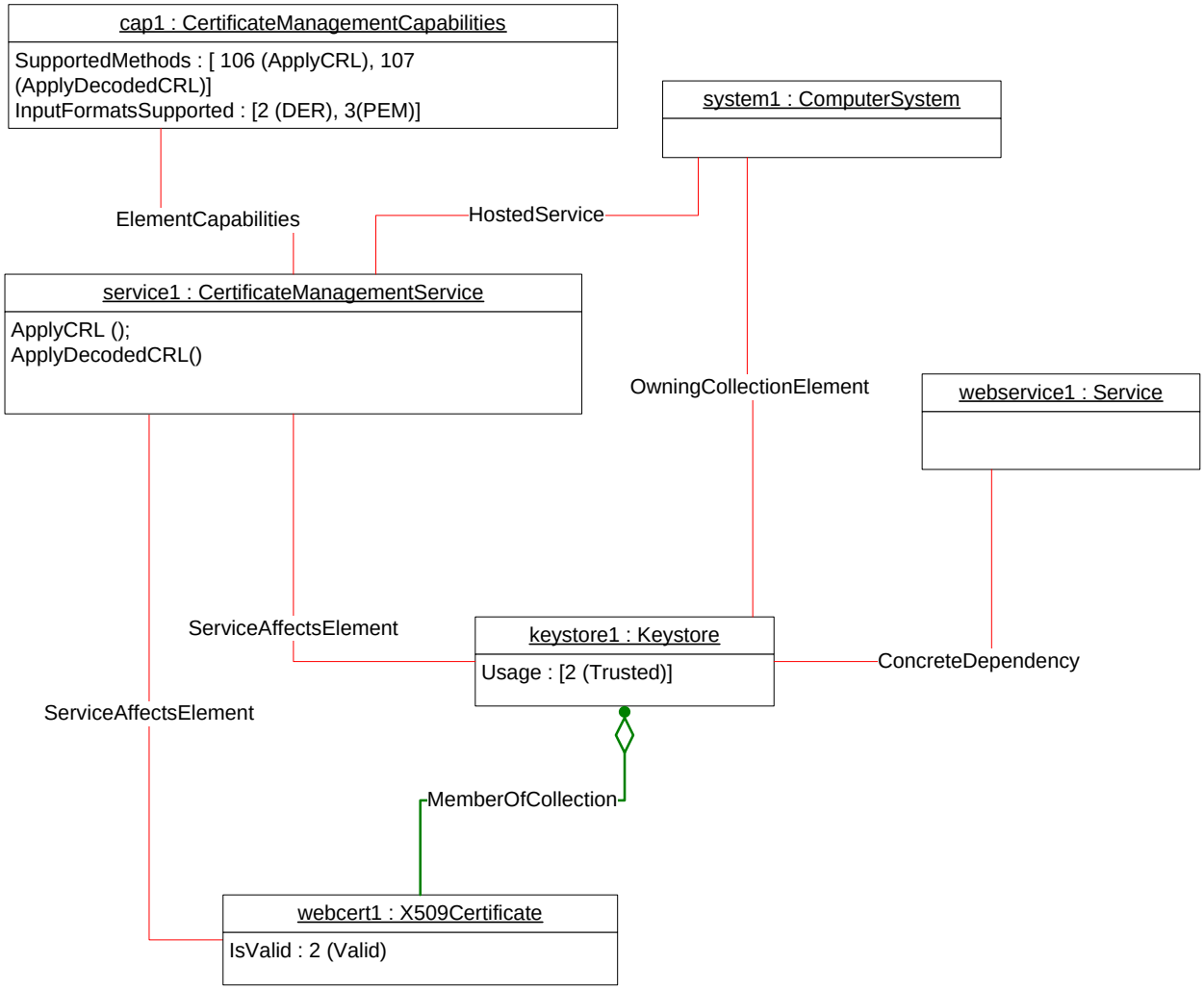
903 **Figure 7 – Import and export of certificates**

904 9.7 CRL

905 Figure 8 and Figure 9 represent an instantiation of the *Certificate Management Profile.*

906 9.7.1 Before applying the CRL

907 In this instantiation, the instrumentation advertises the ability to apply CRLs to key stores using the
908 cap1.SupportedMethods property, which contains the value 106 (ApplyCRL). The
909 cap1.InputFormatsSupported property advertises the capability of service1 to import CRLs in the
910 specified formats, and the value of the Format parameter on the ApplyCRL method will match one of the
911 values in this property array. The EncodedCRL[] parameter contains the encoded CRL, which consists of
912 serial numbers of revoked certificates. In this case, one of these certificates is represented by webcert1.



913

914 **Figure 8 – Before CRL application**

915 ### 9.7.2 After Applying the CRL

916 Upon successful invocation of the ImportEncodedCertificates method a new instance of CIM_X509CRL,
917 webCRL1, will be instantiated and the reference of the instance returned as the NewCertificates[] output
918 parameter. The association instances referencing webCRL1 are created based on the other parameters
919 of the invoked method. (See 8.8 and the method descriptions in the CIM schema.) Upon successful
920 execution, the certificate represented by webcert1 will be revoked and the webcert1.IsValid property will
921 be set to 3 (Invalid).

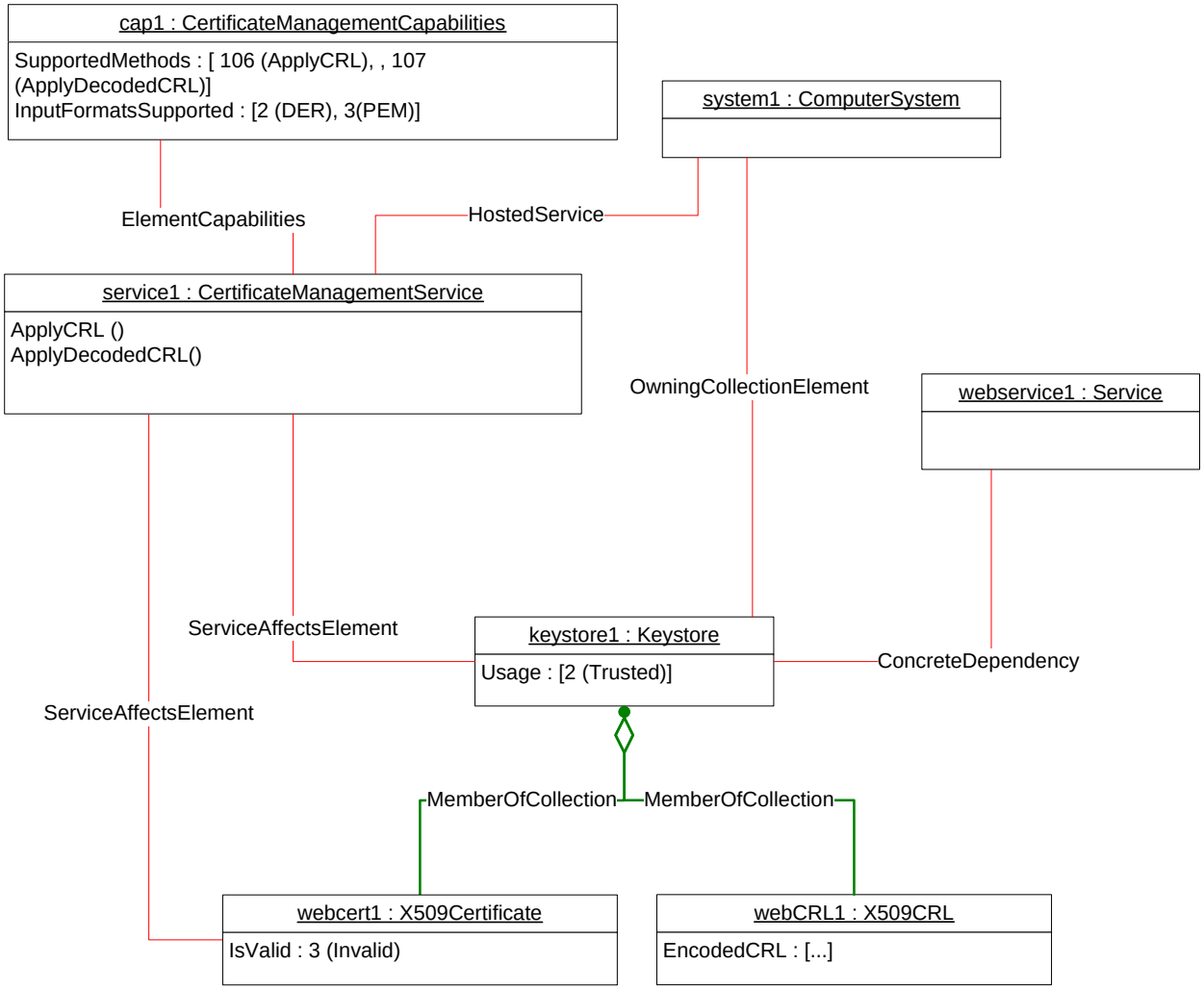


Figure 9 – After CRL application

10 CIM elements

Table 33 shows the instances of CIM elements for this profile. Instances of the CIM elements shall be implemented as described in Table 33. Clauses 7 ("Implementation") and 8 ("Methods") may impose additional requirements on these elements.

Table 33 – CIM elements: Certificate Management Profile

Element Name	Requirement	Description
Classes		
CIM_AssociatedPrivilege	Optional	See 10.18 and 7.4.
CIM_CertificateManagementCapabilities	Mandatory	See 10.1 and 7.2.2.
CIM_CertificateManagementService	Mandatory	See 10.2 and 7.2.
CIM_CredentialContext	Optional	See 10.3.
CIM_ConcreteDependency (CIM_Keystore)	Optional	See 10.4.

Element Name	Requirement	Description
CIM_ConcreteDependency (CIM_X509Certificate)	Optional	See 10.5.
CIM_ConcreteDependency	Optional	See 10.6
CIM_ElementCapabilities	Mandatory	See 10.7.
CIM_HostedService	Mandatory	See 10.8.
CIM_Keystore	Optional	See 7.1 and 10.9.
CIM_ServiceAffectsElement (CIM_Credential)	Optional	See 10.10.
CIM_MemberOfCollection	Conditional	See 10.11.
CIM_OwningCollectionElement	Conditional	See 10.12.
CIM_RegisteredProfile	Mandatory	See 10.13.
CIM_ServiceAffectsElement (CIM_Keystore)	Conditional	See 10.14.
CIM_UnsignedCredential	Mandatory	Zero or more instances shall be implemented. See 10.15.
CIM_X509Certificate	Mandatory	Zero or more instances shall be implemented. See 7.3 and 10.16.
CIM_X509CRL	Mandatory	Zero or more instances shall be implemented. See 10.17.
Indications		
None defined in this profile		

929 ## 10.1 CIM_CertificateManagementCapabilities

930 CIM_CertificateManagementCapabilities is used to represent the capabilities of the
931 CIM_CertificateManagementService. Table 34 details the requirements for instances of
932 CIM_CertificateManagementCapabilities.

933 **Table 34 – Class: CIM_CertificateManagementCapabilities**

Elements	Requirement	Notes
InstanceID	Mandatory	**Key**
ElementName	Mandatory	Pattern ".*"
SupportedMethods	Mandatory	None
AsymmetricKeyGeneration	Mandatory	None
KeyAlgorithmSupported	Mandatory	None
InputFormatsSupported	Conditional	The property shall be supported if the SupportedMethods property has the values 103 (ImportEncodedCertificates) and 106 (ApplyCRL).

Elements	Requirement	Notes
OutputFormatsSupported	Conditional	The property shall be supported if the SupportedMethods property contains 105 (ExportEncodedCertificate).
SupportedSignatureAlgorithms	Optional	None
CumulativePrivilegeMethodology	Optional	None

10.2 CIM_CertificateManagementService

CIM_CertificateManagementService is used to manage PKI-based credentials represented by PKI Credential Instances. Table 35 details the requirements for instances of CIM_CertificateManagementService.

Table 35 – Class: CIM_CertificateManagementService

Elements	Requirement	Notes
SystemCreationClassName	Mandatory	**Key**
CreationClassName	Mandatory	**Key**
SystemName	Mandatory	**Key**
Name	Mandatory	**Key**
ElementName	Mandatory	Pattern ".*"
ImportPublicPrivateKeyPair( )	Conditional	See 8.1.
CreateKeystore( )	Conditional	See 8.2.
CreateCertificateSigningRequest( )	Conditional	See 8.3.
CreateSelfSignedCertificate( )	Conditional	See 8.4.
ImportEncodedCertificates( )	Conditional	See 8.5 .
ImportCertificates( )	Conditional	See 8.6.
ExportEncodedCertificates( )	Conditional	See 8.7.
ApplyCRL( )	Conditional	See 8.8.
ApplyDecodedCRL ( )	Conditional	See 8.9.

10.3 CIM_CredentialContext

CIM_CredentialContext is used to associate the PKI Credential Instance with the instances of subclasses of CIM_ManagedElement that represent the managed element that utilizes the credential. Table 36 details the requirements for instances of CIM_CredentialContext.

943 **Table 36 – Class: CIM_CredentialContext**

Elements	Requirement	Notes
ElementInContext	Mandatory	**Key:** This property shall be a reference to the PKI Credential Instance. Cardinality * indicating zero or more references
ElementProvidingContext	Mandatory	**Key:** This property shall be a reference to the instance of the subclass of CIM_ManagedElement. Cardinality * indicating zero or more references

944 ## 10.4 CIM_ConcreteDependency (CIM_Keystore)

945 CIM_ConcreteDependency is used to associate instances of a CIM_ManagedElement subclass with an
946 instance of CIM_Keystore that the managed element utilizes. Table 37 details the requirements for
947 instances of CIM_ConcreteDependency.

948 **Table 37 – Class: CIM_ConcreteDependency**

Elements	Requirement	Notes
Antecedent	Mandatory	**Key:** This property shall be a reference to the CIM_ManagedElement. Cardinality * indicating zero or more references
Dependent	Mandatory	**Key:** This property shall be a reference to the CIM_Keystore. Cardinality * indicating zero or more references

949 ## 10.5 CIM_ConcreteDependency (CIM_X509Certificate)

950 CIM_ConcreteDependency is used to associate an instance of CIM_X509Certificate with another
951 instance of CIM_X509Certificate to show the certificate chain hierarchy. Table 38 details the requirements
952 for instances of CIM_ConcreteDependency.

953 **Table 38 – Class: CIM_ConcreteDependency**

Elements	Requirement	Notes
Antecedent	Mandatory	**Key:** This property shall be a reference to the CIM_X509Certificate instance representing the signing certificate. Cardinality 0..1 indicating zero or one reference
Dependent	Mandatory	**Key:** This property shall be a reference to the CIM_X509Certificate instance representing the signed certificate. Cardinality * indicating zero or more references

954 ## 10.6 CIM_ConcreteDependency

955 CIM_ConcreteDependency is used to associate an instance of CIM_UnsignedCredential with an instance
956 of CIM_X509Certificate that was generated based on the imported CIM_UnsignedCredential instance
957 representing public-private key pair. Table 40 details the requirements for instances of
958 CIM_ConcreteDependency.

959 **Table 39 – CIM_ConcreteDependency**

Properties	Requirement	Notes
Antecedent	Mandatory	**Key**: Shall reference the instance of CIM_UnsignedCredential Cardinality 0..1 indicating zero or one reference
Dependent	Mandatory	**Key**: Shall reference the instance of CIM_X509Certificate Cardinality * indicating zero or more references

960 ## 10.7 CIM_ElementCapabilities

961 CIM_ElementCapabilities is used to associate an instance of CIM_CertificateManagementService with an
962 instance of CIM_CertificateManagementCapabilities that describes the capabilities of the service. Table
963 40 details the requirements for instances of CIM_ElementCapabilities.

964 **Table 40 – CIM_ElementCapabilities**

Properties	Requirement	Notes
ManagedElement	Mandatory	**Key**: Shall reference the instance of CIM_CertificateManagementService Cardinality 1..* indicating one or more references
Capabilities	Mandatory	**Key**: Shall reference the instance of CIM_CertificateManagementCapabilities Cardinality 1 indicating one and only one reference

965 ## 10.8 CIM_HostedService

966 CIM_HostedService is used to associate CIM_CertificateManagementService with the Scoping Class.
967 Table 41 details the requirements for instances of CIM_HostedService.

968 **Table 41 – Class: CIM_HostedService**

Elements	Requirement	Notes
Antecedent	Mandatory	**Key:** This property shall be a reference to the Scoping Instance. Cardinality 1 indicating one and only one reference
Dependent	Mandatory	**Key:** This property shall be a reference to the Central Instance. Cardinality 1..* indicating one or more references

969 ## 10.9 CIM_Keystore

970 CIM_Keystore is used to represent the key store that accumulates PKI-based credentials represented by
971 PKI Credential Instances. Table 42 details the requirements for instances of CIM_Keystore.

972 **Table 42 – Class: CIM_Keystore**

Elements	Requirement	Notes
InstanceID	Mandatory	**Key**
ElementName	Mandatory	Pattern ".*"

Elements	Requirement	Notes
Usage	Mandatory	None

10.10 CIM_ServiceAffectsElement (CIM_Credential)

CIM_ServiceAffectsElement is used to associate an instance of CIM_CertificateManagementService with an instance of CIM_X509Certificate, CIM_X509CRL, or CIM_UnsignedCredential. If the CIM_Keystore instance exists, CIM_ServiceAffectsElement is optional.

Table 43 contains the requirements for elements of this class.

Table 43 – Class: CIM_ServiceAffectsElement (CIM_Credential)

Elements	Requirement	Notes
AffectedElement	Mandatory	**Key**: This property shall reference the instance of CIM_X509Certificate, CIM_X509CRL, or CIM_UnsignedCredential. Cardinality * indicating zero or more references
AffectingElement	Mandatory	**Key**: This property shall reference the instance of CIM_CertificateManagementService. Cardinality 1 indicating one reference
ElementAffects	Mandatory	Matches 5 (Manages)

10.11 CIM_MemberOfCollection

CIM_MemberOfCollection is used to associate PKI Credential Instances with the instance of CIM_Keystore representing the key store that accumulates the PKI-based credentials. If the PKI Credential Instance exists, the CIM_MemberOfCollection instance is mandatory.

Table 44 provides information about the properties of CIM_MemberOfCollection.

Table 44 – Class: CIM_MemberOfCollection

Properties	Requirement	Notes
GroupComponent	Mandatory	**Key:** This property shall reference an instance of CIM_Keystore. Cardinality 1 indicating one and only one reference
PartComponent	Mandatory	**Key:** This property shall reference PKI Credential Instances. Cardinality * indicating zero or more references

10.12 CIM_OwningCollectionElement

CIM_OwningCollectionElement is used to associate a CIM_Keystore instance with its scoping CIM_System instance. If the CIM_Keystore instance exists, the CIM_OwningCollectionElement is mandatory.

Table 45 provides information about the properties of CIM_OwningCollectionElement.

Table 45 – Class: CIM_OwningCollectionElement

Properties	Requirement	Notes
OwningElement	Mandatory	**Key:** This property shall reference the Scoping Instance of this profile. Cardinality 1 indicating one and only one reference
OwnedElement	Mandatory	**Key:** This property shall be an instance of CIM_Keystore. Cardinality * indicating zero or more references

10.13 CIM_RegisteredProfile

The CIM_RegisteredProfile class is defined by the *Profile Registration Profile*. The requirements denoted in Table 46 are in addition to those mandated by the *Profile Registration Profile*.

Table 46 – Class: CIM_RegisteredProfile

Elements	Requirement	Notes
RegisteredName	Mandatory	This property shall have a value of “Certificate Management”.
RegisteredVersion	Mandatory	This property shall have a value of “1.0.0”.
RegisteredOrganization	Mandatory	This property shall have a value of 2 (“DMTF”).

10.14 CIM_ServiceAffectsElement (CIM_Keystore)

CIM_ServiceAffectsElement is used to associate an instance of CIM_CertificateManagementService with an instance of CIM_Keystore that represents a key store that could be managed using the service. If the CIM_Keystore instance exists, CIM_ServiceAffectsElement is mandatory.

Table 47 contains the requirements for elements of this class.

1000 **Table 47 – Class: CIM_ServiceAffectsElement (CIM_Keystore)**

Elements	Requirement	Notes
AffectedElement	Mandatory	**Key**: This property shall reference the instance of CIM_Keystore. Cardinality * indicating zero or more references
AffectingElement	Mandatory	**Key**: This property shall reference the instance of CIM_CertificateManagementService. Cardinality 1 indicating one reference.
ElementAffects	Mandatory	Matches 5 (Manages)

1001 ## 10.15 CIM_UnsignedCredential

1002 CIM_UnsignedCredential is used to represent the Asymmetric key (public/private key pair). Table 48
1003 details the requirements for instances of CIM_UnsignedCredential.

1004 **Table 48 – Class: CIM_UnsignedCredential**

Elements	Requirement	Notes
InstanceID	Mandatory	**Key**
ElementName	Mandatory	Pattern ".*"
PublicKey	Mandatory	

1005 ## 10.16 CIM_X509Certificate

1006 CIM_X509Certificate is used to represent the X509 certificate. Table 49 details the requirements for
1007 instances of CIM_X509Certificate.

1008 **Table 49 – Class: CIM_X509Certificate**

Elements	Requirement	Notes
InstanceID	Mandatory	**Key**
ElementName	Mandatory	Pattern ".*"
TrustedRootCertificate	Optional	
Expires	Optional	
ValidFrom	Optional	
IsValid	Optional	
EncodedCertificate	Mandatory	

10.17 CIM_X509CRL

CIM_X509CRL is used to represent the X509 CRL. Table 50 details the requirements for instances of CIM_X509CRL.

Table 50 – Class: CIM_X509CRL

Elements	Requirement	Notes
InstanceID	Mandatory	**Key**
ElementName	Mandatory	Pattern ".*"
Issued	Optional	
NextUpdate	Optional	
Issuer	Optional	If the SupportedMethods property array of the associated instance of CIM_CertificateManagementCapabilities contains the value 107 (ApplyDecodedCRL), the Issuer property shall be supported.
SerialNumbers	Conditional	If the SupportedMethods property array of the associated instance of CIM_CertificateManagementCapabilities contains the value 107 (ApplyDecodedCRL), the Issuer property shall be supported.
EncodedCRL	Conditional	If the SupportedMethods property array of the associated instance of CIM_CertificateManagementCapabilities contains the value 106 (ApplyCRL), the Issuer property shall be supported.

10.18 CIM_AssociatedPrivilege

CIM_AssociatedPrivilege is used to associate the security principal with the credential or credential store to which the security principal has access authorization.

Table 51 lists requirements that are in addition to those enumerated in the *Credential Management Profile*.

Table 51 – Class: CIM_AssociatedPrivilege

Elements	Requirement	Notes
Activities	Mandatory	See clause 7.4.

ANNEX A
(informative)

Change Log

Version	Date	Description
1.0.0	2011-09-16	DMTF Standard