[MS-ICPR-Diff]:

ICertPassage Remote Protocol

Intellectual Property Rights Notice for Open Specifications Documentation

- **Technical Documentation.** Microsoft publishes Open Specifications documentation ("this documentation") for protocols, file formats, data portability, computer languages, and standards ~~as well as overviews of the interaction among each of these technologies~~support. Additionally, overview documents cover inter-protocol relationships and interactions.
- **Copyrights**. This documentation is covered by Microsoft copyrights. Regardless of any other terms that are contained in the terms of use for the Microsoft website that hosts this documentation, you ~~may~~can make copies of it in order to develop implementations of the technologies that are described in ~~the Open Specifications~~ this documentation and ~~may~~can distribute portions of it in your implementations ~~using~~that use these technologies or in your documentation as necessary to properly document the implementation. You ~~may~~can also distribute in your implementation, with or without modification, any ~~schema, IDL's~~schemas, IDLs, or code samples that are included in the documentation. This permission also applies to any documents that are referenced in the Open Specifications~~.~~ documentation.
- **No Trade Secrets**. Microsoft does not claim any trade secret rights in this documentation.
- **Patents**. Microsoft has patents that ~~may~~might cover your implementations of the technologies described in the Open Specifications~~.~~ documentation. Neither this notice nor Microsoft's delivery of ~~the~~this documentation grants any licenses under those patents or any other Microsoft patents. However, a given Open ~~Specification may~~Specifications document might be covered by the Microsoft Open Specifications Promise or the Microsoft Community Promise. If you would prefer a written license, or if the technologies described in ~~the Open Specifications~~this documentation are not covered by the Open Specifications Promise or Community Promise, as applicable, patent licenses are available by contacting iplg@microsoft.com.
- **Trademarks**. The names of companies and products contained in this documentation ~~may~~might be covered by trademarks or similar intellectual property rights. This notice does not grant any licenses under those rights. For a list of Microsoft trademarks, visit www.microsoft.com/trademarks.
- **Fictitious Names**. The example companies, organizations, products, domain names, ~~e-mail~~email addresses, logos, people, places, and events that are depicted in this documentation are fictitious. No association with any real company, organization, product, domain name, email address, logo, person, place, or event is intended or should be inferred.

Reservation of Rights. All other rights are reserved, and this notice does not grant any rights other than as specifically described above, whether by implication, estoppel, or otherwise.

Tools. The Open Specifications ~~do~~documentation does not require the use of Microsoft programming tools or programming environments in order for you to develop an implementation. If you have access to Microsoft programming tools and environments, you are free to take advantage of them. Certain Open Specifications documents are intended for use in conjunction with publicly available ~~standard~~standards specifications and network programming art~~,~~ and ~~assumes~~, as such, assume that the reader either is familiar with the aforementioned material or has immediate access to it.

Revision Summary

Date	Revision History	Revision Class	Comments
3/2/2007	1.0	New	Version 1.0 release
4/3/2007	1.1	Minor	Version 1.1 release
5/11/2007	1.2	Minor	Version 1.2 release
6/1/2007	1.2.1	Editorial	Changed language and formatting in the technical content.
7/3/2007	1.2.2	Editorial	Changed language and formatting in the technical content.
8/10/2007	2.0	Major	IDL was updated.
9/28/2007	2.0.1	Editorial	Changed language and formatting in the technical content.
10/23/2007	3.0	Major	Converted document to unified format and updated technical content.
1/25/2008	3.0.1	Editorial	Changed language and formatting in the technical content.
3/14/2008	3.0.2	Editorial	Changed language and formatting in the technical content.
6/20/2008	4.0	Major	Updated and revised the technical content.
7/25/2008	4.0.1	Editorial	Changed language and formatting in the technical content.
8/29/2008	4.0.2	Editorial	Fix capitalization issues.
10/24/2008	4.0.3	Editorial	Changed language and formatting in the technical content.
12/5/2008	4.0.4	Editorial	Editorial Update.
1/16/2009	4.0.5	Editorial	Changed language and formatting in the technical content.
2/27/2009	4.0.6	Editorial	Changed language and formatting in the technical content.
4/10/2009	5.0	Major	Updated and revised the technical content.
5/22/2009	5.0.1	Editorial	Changed language and formatting in the technical content.
7/2/2009	6.0	Major	Updated and revised the technical content.
8/14/2009	6.0.1	Editorial	Changed language and formatting in the technical content.
9/25/2009	6.1	Minor	Clarified the meaning of the technical content.
11/6/2009	6.1.1	Editorial	Changed language and formatting in the technical content.
12/18/2009	6.1.2	Editorial	Changed language and formatting in the technical content.
1/29/2010	7.0	Major	Updated and revised the technical content.
3/12/2010	8.0	Major	Updated and revised the technical content.
4/23/2010	9.0	Major	Updated and revised the technical content.
6/4/2010	10.0	Major	Updated and revised the technical content.
7/16/2010	11.0	Major	Updated and revised the technical content.

Date	Revision History	Revision Class	Comments
8/27/2010	12.0	Major	Updated and revised the technical content.
10/8/2010	12.1	Minor	Clarified the meaning of the technical content.
11/19/2010	13.0	Major	Updated and revised the technical content.
1/7/2011	13.0	None	No changes to the meaning, language, or formatting of the technical content.
2/11/2011	13.0	None	No changes to the meaning, language, or formatting of the technical content.
3/25/2011	14.0	Major	Updated and revised the technical content.
5/6/2011	14.0	None	No changes to the meaning, language, or formatting of the technical content.
6/17/2011	14.1	Minor	Clarified the meaning of the technical content.
9/23/2011	15.0	Major	Updated and revised the technical content.
12/16/2011	16.0	Major	Updated and revised the technical content.
3/30/2012	16.0	None	No changes to the meaning, language, or formatting of the technical content.
7/12/2012	16.1	Minor	Clarified the meaning of the technical content.
10/25/2012	16.1	None	No changes to the meaning, language, or formatting of the technical content.
1/31/2013	16.1	None	No changes to the meaning, language, or formatting of the technical content.
8/8/2013	17.0	Major	Updated and revised the technical content.
11/14/2013	17.0	None	No changes to the meaning, language, or formatting of the technical content.
2/13/2014	17.0	None	No changes to the meaning, language, or formatting of the technical content.
5/15/2014	17.0	None	No changes to the meaning, language, or formatting of the technical content.
6/30/2015	18.0	Major	Significantly changed the technical content.
10/16/2015	18.0	~~No Change~~None	No changes to the meaning, language, or formatting of the technical content.

Table of Contents

1 Introduction

This document specifies the ICertPassage Remote Protocol. This protocol is a subset of the Windows Client Certificate Enrollment Protocol, as specified in [MS-WCCE]. The difference between this protocol and the Windows Client Certificate Enrollment Protocol is that this protocol only allows the **client** to **enroll certificates**, whereas the Windows Client Certificate Enrollment Protocol provides **enrollment** and additional functionality, such as the capability to read **certification authority (CA)** data and configuration information. Reading and understanding the Windows Client Certificate Enrollment Protocol, as specified in [MS-WCCE], is essential to understanding the ICertPassage Remote Protocol.

Sections 1.5, 1.8, 1.9, 2, and 3 of this specification are normative ~~and can contain the terms MAY, SHOULD, MUST, MUST NOT, and SHOULD NOT as defined in [RFC2119]. Sections 1.5 and 1.9 are also normative but do not contain those terms~~. All other sections and examples in this specification are informative.

1.1 Glossary

~~The~~This document uses the following terms ~~are specific to this document~~:

Active Directory: A general-purpose network directory service. **Active Directory** also refers to the Windows implementation of a directory service. **Active Directory** stores information about a variety of objects in the network. Importantly, user accounts, computer accounts, groups, and all related credential information used by the Windows implementation of Kerberos are stored in **Active Directory**. **Active Directory** is either deployed as Active Directory Domain Services (AD DS) or Active Directory Lightweight Directory Services (AD LDS). [MS-ADTS] describes both forms. For more information, see [MS-AUTHSOD] section 1.1.1.5.2, Lightweight Directory Access Protocol (LDAP) versions 2 and 3, Kerberos, and DNS.

certificate: A certificate is a collection of attributes (1) and extensions that can be stored persistently. The set of attributes in a certificate can vary depending on the intended usage of the certificate. A certificate securely binds a public key to the entity that holds the corresponding private key. A certificate is commonly used for authentication (2) and secure exchange of information on open networks, such as the Internet, extranets, and intranets. Certificates are digitally signed by the issuing **certification authority (CA)** and can be issued for a user, a computer, or a service. The most widely accepted format for certificates is defined by the ITU-T X.509 version 3 international standards. For more information about attributes and extensions, see [RFC3280] and [X509] sections 7 and 8.

certificate enrollment: The process of acquiring a digital certificate from a **certificate authority (CA)**, which typically requires an end entity to first makes itself known to the CA (either directly, or through a registration authority). This certificate and its associated **private key** establish a trusted identity for an entity that is using the **public key**–based services and applications. Also referred to as simply "enrollment".

certification authority (CA): A third party that issues **public key certificates**. Certificates serve to bind public keys to a user identity. Each user and certification authority (CA) can decide whether to trust another user or CA for a specific purpose, and whether this trust should be transitive. For more information, see [RFC3280].

client: A computer on which the remote procedure call (RPC) client is executing.

digital signature: A value that is generated by using a digital signature algorithm, taking as input a private key and an arbitrary-length string, such that a specific verification algorithm is satisfied by the value, the input string, and the public key corresponding to the input private key.

Distributed Component Object Model (DCOM): The Microsoft Component Object Model (COM) specification that defines how components communicate over networks, as specified in [MS-DCOM].

dynamic endpoint: A network-specific server address that is requested and assigned at run time. For more information, see [C706].

endpoint: A network-specific address of a remote procedure call (RPC) server process for remote procedure calls. The actual name and type of the endpoint depends on the **RPC** protocol sequence that is being used. For example, for RPC over TCP (RPC Protocol Sequence ncacn_ip_tcp), an endpoint might be TCP port 1025. For RPC over Server Message Block (RPC Protocol Sequence ncacn_np), an endpoint might be the name of a named pipe. For more information, see [C706].

endpoint mapper: A service on a remote procedure call (RPC) server that maintains a database of **dynamic endpoints** and allows **clients** to map an interface/object UUID pair to a local **dynamic endpoint**. For more information, see [C706].

~~**enroll/enrollment**: See certification.~~

enroll: To request and acquire a digital certificate from a **certificate authority (CA)**. This is typically accomplished through a **certificate enrollment** process.

private key: One of a pair of keys used in public-key cryptography. The private key is kept secret and is used to decrypt data that has been encrypted with the corresponding public key. For an introduction to this concept, see [CRYPTO] section 1.8 and [IEEE1363] section 3.1.

public key: One of a pair of keys used in public-key cryptography. The public key is distributed freely and published as part of a digital certificate. For an introduction to this concept, see [CRYPTO] section 1.8 and [IEEE1363] section 3.1.

public-private key pair: The association of a public key and its corresponding private key when used in cryptography. Also referred to simply as a "key pair". For an introduction to public-private key pairs, see [IEEE1363] section 3.

remote procedure call (RPC): A context-dependent term commonly overloaded with three meanings. Note that much of the industry literature concerning RPC technologies uses this term interchangeably for any of the three meanings. Following are the three definitions: (*) The runtime environment providing remote procedure call facilities. The preferred usage for this meaning is "RPC runtime". (*) The pattern of request and response message exchange between two parties (typically, a client and a server). The preferred usage for this meaning is "RPC exchange". (*) A single message from an exchange as defined in the previous definition. The preferred usage for this term is "RPC message". For more information about RPC, see [C706].

RPC endpoint: A network-specific address of a server process for remote procedure calls (RPCs). The actual name of the RPC endpoint depends on the RPC protocol sequence being used. For example, for the NCACN_IP_TCP RPC protocol sequence an RPC endpoint might be TCP port 1025. For more information, see [C706].

server: A computer on which the **remote procedure call (RPC)** server is executing.

universally unique identifier (UUID): A 128-bit value. UUIDs can be used for multiple purposes, from tagging objects with an extremely short lifetime, to reliably identifying very persistent objects in cross-process communication such as client and server interfaces, manager entry-point vectors, and **RPC** objects. UUIDs are highly likely to be unique. UUIDs are also known as globally unique identifiers (GUIDs) and these terms are used interchangeably in the Microsoft protocol technical documents (TDs). Interchanging the usage of these terms does not imply or require a specific algorithm or mechanism to generate the UUID. Specifically, the use of this term does not imply or require that the algorithms described in [RFC4122] or [C706] must be used for generating the UUID.

well-known endpoint: A preassigned, network-specific, stable address for a particular client/server instance. For more information, see [C706].

MAY, SHOULD, MUST, SHOULD NOT, MUST NOT: These terms (in all caps) are used as defined in [RFC2119]. All statements of optional behavior use either MAY, SHOULD, or SHOULD NOT.

1.2 References

Links to a document in the Microsoft Open Specifications library point to the correct section in the most recently published version of the referenced document. However, because individual documents in the library are not updated at the same time, the section numbers in the documents may not match. You can confirm the correct section numbering by checking the Errata.

1.2.1 Normative References

We conduct frequent surveys of the normative references to assure their continued availability. If you have any issue with finding a normative reference, please contact dochelp@microsoft.com. We will assist you in finding the relevant information.

[C706] The Open Group, "DCE 1.1: Remote Procedure Call", C706, August 1997, https://www2.opengroup.org/ogsys/catalog/c706

[MS-CRTD] Microsoft Corporation, "Certificate Templates Structure".

[MS-CSRA] Microsoft Corporation, "Certificate Services Remote Administration Protocol".

[MS-DCOM] Microsoft Corporation, "Distributed Component Object Model (DCOM) Remote Protocol".

[MS-DTYP] Microsoft Corporation, "Windows Data Types".

[MS-RPCE] Microsoft Corporation, "Remote Procedure Call Protocol Extensions".

[MS-WCCE] Microsoft Corporation, "Windows Client Certificate Enrollment Protocol".

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997, http://www.rfc-editor.org/rfc/rfc2119.txt

[RFC2797] Myers, M., Liu, X., Schaad, J., and Weinstein, J., "Certificate Management Messages Over CMS", RFC 2797, April 2000, http://www.ietf.org/rfc/rfc2797.txt

[RFC2986] Nystrom, M. and Kaliski, B., "PKCS#10: Certificate Request Syntax Specification", RFC 2986, November 2000, http://www.ietf.org/rfc/rfc2986.txt

[RFC3280] Housley, R., Polk, W., Ford, W., and Solo, D., "Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile", RFC 3280, April 2002, http://www.ietf.org/rfc/rfc3280.txt

[RFC3852] Housley, R., "Cryptographic Message Syntax (CMS)", RFC 3852, July 2004, http://www.ietf.org/rfc/rfc3852.txt

[UNICODE4.0] The Unicode Consortium, "Unicode 4.0.0", http://www.unicode.org/versions/Unicode4.0.0/

[X509] ITU-T, "Information Technology - Open Systems Interconnection - The Directory: Public-Key and Attribute Certificate Frameworks", Recommendation X.509, August 2005, http://www.itu.int/rec/T-REC-X.509/en

[X660] ITU-T, "Information Technology - Open Systems Interconnection - Procedures for the Operation of OSI Registration Authorities: General Procedures and Top Arcs of the ASN.1 Object Identifier Tree", Recommendation X.660, August 2004, http://www.itu.int/rec/T-REC-X.660/en

[X690] ITU-T, "Information Technology - ASN.1 Encoding Rules: Specification of Basic Encoding Rules (BER), Canonical Encoding Rules (CER) and Distinguished Encoding Rules (DER)", Recommendation X.690, July 2002, http://www.itu.int/rec/T-REC-X.690/en

1.2.2 Informative References

None.

1.3 Overview

The ICertPassage Remote Protocol exposes a **Remote Procedure Call (RPC)** (as specified in [MS-RPCE]) interface that allows a client to interact with a certification authority (CA) to request and receive X.509 certificates (as specified in [X509]) from the CA. The ICertPassage Remote Protocol only provides certificate enrollment functionality. The Windows Client Certificate Enrollment Protocol (as specified in [MS-WCCE]) provides a larger set of functionality, including reading CA data and configuration information. The certificate enrollment process and protocol overview are as specified in [MS-WCCE] section 1.3.

The ICertPassage interface defines one method: CertServerRequest (section 3.2.4.1.1).

1.4 Relationship to Other Protocols

The ICertPassage Remote Protocol depends on the Remote Procedure Call Protocol Extensions, as specified in [MS-RPCE]. No other Windows protocol depends on the ICertPassage Remote Protocol. The following diagram shows the layering of the protocol stack.

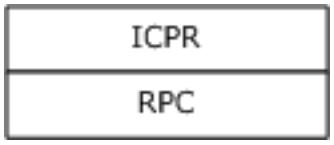

Figure 1: ICRP Protocol Stack

The ICertPassage Remote Protocol shares ADM elements with Windows Client Certificate Enrollment Protocol [MS-WCCE] as specified in section 3.1.1 and section 3.2.1. The ICertPassage Remote Protocol, the Certificate Services Remote Administration Protocol, and the Windows Client Certificate Enrollment Protocol use a common list of configuration data elements, defined in [MS-WCCE] section 3.2.1.1.4.

1.5 Prerequisites and Preconditions

The ICertPassage Remote Protocol has the same prerequisites as the Windows Client Certificate Enrollment Protocol, as specified in [MS-WCCE] section 1.5.

ICertPassage Remote Protocol **server** implementations that also implement the Certificate Services Remote Administration Protocol specified in [MS-CSRA] or the Windows Client Certificate Enrollment Protocol specified in [MS-WCCE] use the same configuration data elements, defined in [MS-WCCE] section 3.2.1.1.4 as "public", for those implementations.

1.6 Applicability Statement

This protocol applies to legacy clients that ~~must~~have to use RPC (as specified in [MS-RPCE]) to interact with a CA for the purpose of enrolling or managing X.509 (as specified in [X509]) certificates.

If clients can interact with the CA through **Distributed Component Object Model (DCOM)** (as specified in [MS-DCOM]) interfaces, they ~~should~~are to use the Windows Client Certificate Enrollment Protocol, as specified in [MS-WCCE].

1.7 Versioning and Capability Negotiation

Version and capability negotiation is not provided in this protocol.<1>

1.8 Vendor-Extensible Fields

None.

1.9 Standards Assignments

None.

2 Messages

The following sections specify how ICertPassage Remote Protocol messages are transported and ICertPassage Remote Protocol message syntax.

2.1 Transport

This protocol uses the following RPC protocol sequence: RPC over named pipe and RPC over TCP/IP, as specified in [MS-RPCE].

The **endpoint** pipe name for RPC over named pipe, as specified in [MS-RPCE], is `\PIPE\cert`. This endpoint is used for the authenticated RPC interface. The authenticated RPC interface allows RPC to negotiate the use of authentication and the authentication level on behalf of the client and server, as specified in [MS-RPCE].

In the case of using RPC over TCP, this protocol uses RPC **dynamic endpoints** as defined in Part 4 of [C706]. For more information, see Client Initialization (section 3.1.3) and Server Initialization (section 3.2.3).

This protocol MUST use the **universal unique identifier (UUID)**, as specified in section 3.2.4.1.

2.2 Common Data Types

The ICertPassage interface uses the CERTTRANSBLOB structure, as specified in [MS-WCCE] section 2.2.2.2.

This protocol specification makes use of the ~~BYTE,~~ wchar_t~~,~~ and DWORD datatypes defined in [MS-DTYP] sections 2.~~2.6, 2.~~1.6~~,~~ and 2.2.9.

2.2.1 Request Format

The ICertPassage Remote Protocol is a simple request-response pattern between the client and the server. The client MUST send the certificate request using one of the following ASN.1 DER encoded message formats:

- PKCS #10 as specified in [RFC2986].
- Cryptographic Message Syntax (CMS) as specified in [RFC3852].
- Certificate Management Messages over CMS (CMC) as specified in [RFC2797].

Details are as specified in [MS-WCCE] section 2.2.2.6. Each format contains a set of attributes and extensions describing the request.<2>

2.2.2 Response Format

Responses are returned by the ICertPassage Remote Protocol in either CMS format or CMC format. Details are as specified in [MS-WCCE] section 2.2.2.8. The format of the response is determined by the value passed in the *dwFlags* parameter, as specified in section 3.1.4.1.

3 Protocol Details

The ICertPassage Remote Protocol is a simple request-response protocol. The client sends a certificate request, and the server responds with a signed certificate or a detailed disposition message. In almost all cases, the protocol is a single message followed by a single reply. Details on the flow and sequencing of the certificate enrollment protocol are as specified in [MS-WCCE] section 3.

3.1 ICertPassage Client Details

Details of the client role are exactly as specified in [MS-WCCE] section 3.1.

3.1.1 Abstract Data Model

The client abstract data model is as specified in the abstract data model subsections of the [MS-WCCE] section 3.1.

3.1.2 Timers

None.

3.1.3 Initialization

The client creates an RPC association (or binding) to the server **RPC endpoint** (as specified in section 2.1) when an RPC method is called. The client SHOULD create a separate association for each method invocation, or it MAY reuse an association for multiple invocations.

The client SHOULD create an authenticated RPC association with the highest possible authentication level. RPC authentication levels are as specified in [MS-RPCE].<3> Because the RPC server endpoint is dynamic, the client MUST use the RPC **endpoint mapper** services (as specified in [MS-RPCE] section 2.2.1.2) to locate the endpoint at which the server is registered.

3.1.4 Message Processing and Sequencing Rules

3.1.4.1 Processing ICertPassage:: CertServerRequest

Details of the client processing rules are exactly as specified in [MS-WCCE] section 3.1.1.4.3.

3.1.5 Timer Events

None.

3.1.6 Other Local Events

The ICertPassage interface CertServerRequest method is invoked to obtain certificates whenever they are required by the client.

3.2 ICertPassage Server Details

Details of the server processing rules are exactly as specified in [MS-WCCE] section 3.2.

3.2.1 Abstract Data Model

As specified in [MS-WCCE] section 3.2.1.1.

ICertPassage Remote Protocol server implementations that also implement the Certificate Services Remote Administration Protocol specified in [MS-CSRA] or the Windows Client Certificate Enrollment Protocol specified in [MS-WCCE] use the same configuration data elements, defined in [MS-WCCE] section 3.2.1.1.4 as "public", for those implementations. If either Certificate Services Remote Administration Protocol or Windows Client Certificate Enrollment Protocol or both are also implemented, access to the configuration data elements from either or both of these protocols SHOULD be serialized.

3.2.2 Timers

None.

3.2.3 Initialization

Interface initialization: The CA MUST listen on the **well-known endpoint** specified for this RPC interface for the RPC over named pipes binding. The CA also MUST register with the RPC endpoint mapper service for the TCP over RPC binding (as specified in [MS-RPCE] section 2.2.1.2). Details are as specified in section 2.1.

Cryptographic initialization: The CA SHOULD obtain the certificates, the signing **private key**, and the exchange private key. The CA also MUST validate the CA signing certificates and its chain. The validation is based on chain validation, as specified in [RFC3280] section 6.<4>

3.2.4 Message Processing and Sequencing Rules

The ICertPassage Remote Protocol defines the following interface:

ICertPassage (section 3.2.4.1): A method that enables a client to request certificates from a certification authority.

3.2.4.1 ICertPassage Interface

The ICertPassage RPC interface permits the client to submit a certificate enrollment request to the CA and receive a signed X.509 certificate (as specified in [X509]) as the response.

The version number for this interface is 0.0. The UUID for this interface is 91ae6020-9e3c-11cf-8d7c-00aa00c091be, as specified in [MS-RPCE].<5>

The interface defines a single method.

Methods in RPC Opnum Order

Method	Description
CertServerRequest	Opnum: 0

3.2.4.1.1 CertServerRequest (Opnum 0)

The CertServerRequest method processes a certificate enrollment request from the client.<6>

```
DWORD CertServerRequest(
  [in] ~~const~~ handle_t h,
  [in] DWORD dwFlags,
  [in, string, unique] const wchar_t* pwszAuthority,
  [in, out, ref] DWORD* pdwRequestId,
```

```
    [out] DWORD* pdwDisposition,
    [in, ref] const CERTTRANSBLOB* pctbAttribs,
    [in, ref] const CERTTRANSBLOB* pctbRequest,
    [out, ref] CERTTRANSBLOB* pctbCert,
    [out, ref] CERTTRANSBLOB* pctbEncodedCert,
    [out, ref] CERTTRANSBLOB* pctbDispositionMessage
);
```

h: A handle retrieved during the RPC bind operation, as specified in [MS-RPCE] section 2.2.2.

dwFlags: The *dwFlags* parameter has identical syntax and semantics to the *dwFlags* parameter specified in [MS-WCCE] section 3.2.1.4.2.1.

pwszAuthority: The *pwszAuthority* parameter has identical syntax and semantics to the *pwszAuthority* parameter specified in [MS-WCCE] section 3.2.1.4.2.1.

pdwRequestId: The *pdwRequestId* parameter has identical syntax and semantics to the *pdwRequestId* parameter specified in [MS-WCCE] section 3.2.1.4.2.1.

pdwDisposition: The *pdwDisposition* parameter has identical syntax and semantics to the *pdwDisposition* parameter specified in [MS-WCCE] section 3.2.1.4.2.1.

pctbAttribs: A pointer to a CERTTRANSBLOB structure, as specified in [MS-WCCE] section 2.2.2.2, where the *pb* field of this structure points to a Unicode (as specified in [UNICODE4.0]) null-terminated string and the *cb* field contains the length of the string, including the NULL-terminated character (in bytes). If the value of the *cb* field ~~doesn't~~does not match the length, in bytes, of the string (including the terminating null character), the CA MUST return the E_INVALIDARG error (0x80070057) to the client. Otherwise, the semantics of the string pointed to by the *pb* field are identical to the *pwszAttributes* parameter specified in [MS-WCCE] section 3.2.1.4.2.1.

pctbRequest: The *pctbRequest* parameter has identical syntax and semantics to the *pctbRequest* parameter, as specified in [MS-WCCE] section 3.2.1.4.2.1.

pctbCert: The *pctbCert* parameter has identical syntax and semantics to the *pctbCertChain* parameter, as specified in [MS-WCCE] section 3.2.1.4.2.1.

pctbEncodedCert: The *pctbEncodedCert* parameter has identical syntax and semantics to the *pctbEncodedCert* parameter, as specified in [MS-WCCE] section 3.2.1.4.2.1.

pctbDispositionMessage: The *pctbDispositionMessage* parameter has identical syntax and semantics to the *pctbDispositionMessage* parameter, as specified in [MS-WCCE] section 3.2.1.4.2.1.

Return Values: The method MUST return ERROR_SUCCESS (0x00000000) on success. This method's return values MUST have identical syntax and semantics to the return values specified in [MS-WCCE] section 3.2.1.4.2.1.

If the ADM element *Config.CA.Interface.Flags* contains the value IF_NORPCICERTREQUEST, the server SHOULD return an error.<7>

If the ADM element *Config.CA.Interface.Flags* contains the value IF_ENFORCEENCRYPTICERTREQUEST and the RPC_C_AUTHN_LEVEL_PKT_PRIVACY authentication level ([MS-RPCE] section 2.2.1.1.8) is not specified on the RPC connection from the client, the CA MUST refuse to establish a connection with the client by returning E_ACCESSDENIED (0x80000009).

Otherwise, the processing rules for the ICertRequestD::Request method ([MS-WCCE] section 3.2.2.6.2.1) apply, except that if the ADM element *Config.CA.Interface.Flags* contains the value IF_NOREMOTEICERTREQUEST, these values are ignored and the request is processed as though the values were absent.

3.2.5 Timer Events

None.

3.2.6 Other Local Events

None.

4 Protocol Examples

A client is typically configured in such a manner that it is able to determine when it requires a certificate. A common scenario is that a user has been instructed to enroll for a certificate that will allow use of a security-enhanced wireless network. After the user invokes the enrollment process, the following sequence of events occurs:

1. The enrollment client queries **Active Directory** for the templates (as specified in [MS-CRTD]) that are available for the specified user. As the resource manager, Active Directory enforces that the user only receives templates that the user has read permissions to access (as specified in [MS-WCCE] section 2.2.2.11).

2. The user selects the template with cn = Client Authentication (as specified in [MS-CRTD] section 2.1). This template includes the client authentication (OID = 1.3.6.1.5.5.7.3.2) as part of its pkiExtendedKeyUsage attribute (as specified in [MS-CRTD] section 2.12).

3. The client then generates a **public-private key pair** and constructs the CMS as specified in [RFC3852] request message (as specified in section 2.2.1), which:

   - Includes a **public key**.
   - Includes a template name (that is, Client Authentication) as a request attribute.
   - Is signed by the private key, as specified in [RFC3852].

4. The client creates an RPC connection with the CA. See section 3.1.3.

5. Using the connection previously mentioned, the client invokes the ICertPassage::CertServerRequest() method (see section 3.2.4.1.1) and, as a result, submits the CMS certificate request constructed in step 3.

6. The CA receives the certificate request from the client.

7. The CA validates the CMS message for **digital signature** validity and ASN.1 structure accuracy (as specified in [X660] and [X690]).

8. The CA constructs a certificate based on the public key provided, the request attributes and extensions, and the template information.

9. The CA signs the certificate with the CA private key and returns the newly created certificate to the caller in the pctbEncodedCert as an out parameter.

5 Security

5.1 Security Considerations for Implementers

Security considerations for implementation of this protocol are as specified in [MS-WCCE] section 5.

5.2 Index of Security Parameters

Security parameter	Section
Authenticated RPC association	Initialization (section 3.1.3)
Cryptographic initialization	Initialization (section 3.2.3)

6 Appendix A: Full IDL

For ease of implementation, the full IDL is provided below, where "ms-dtyp.idl" is the IDL found in [MS-DTYP] Appendix A and "ms-wcce.idl" is the IDL found in [MS-WCCE] Appendix A.

```
// Please refer to [MS-WCCE] for the definition of the
// CERTTRANSBLOB

import "ms-wcce.idl";

// basic type aliases

typedef byte                BYTE;


[
    uuid(91ae6020-9e3c-11cf-8d7c-00aa00c091be),
    pointer_default(unique)
]
interface ICertPassage
{
    DWORD CertServerRequest(
[in]                        handle_t        h,
[in]                        DWORD           dwFlags,
[in, string, unique] const  wchar_t         *pwszAuthority,
[in, out, ref]              DWORD           *pdwRequestId,
[out]                       DWORD           *pdwDisposition,
[in, ref]            const  CERTTRANSBLOB   *pctbAttribs,
[in, ref]            const  CERTTRANSBLOB   *pctbRequest,
[out, ref]                  CERTTRANSBLOB   *pctbCert,
[out, ref]                  CERTTRANSBLOB   *pctbEncodedCert,
[out, ref]                  CERTTRANSBLOB   *pctbDispositionMessage);
}
```

7 Appendix B: Product Behavior

The information in this specification is applicable to the following Microsoft products or supplemental software. References to product versions include released service packs.

~~Note: Some of the information in this section is subject to change because it applies to a preliminary product version, and thus may differ from the final version of the software when released. All behavior notes that pertain to the preliminary product version contain specific references to it as an aid to the reader.~~

- Windows NT operating system
- Windows 2000 operating system
- Windows XP operating system
- Windows Server 2003 operating system
- Windows Vista operating system
- Windows Server 2008 operating system
- Windows 7 operating system
- Windows Server 2008 R2 operating system
- Windows 8 operating system
- Windows Server 2012 operating system
- Windows 8.1 operating system
- Windows Server 2012 R2 operating system
- Windows 10 operating system
- Windows Server 2016 ~~Technical Preview~~ operating system

Exceptions, if any, are noted below. If a service pack or Quick Fix Engineering (QFE) number appears with the product version, behavior changed in that service pack or QFE. The new behavior also applies to subsequent service packs of the product unless otherwise specified. If a product edition appears with the product version, behavior is different in that product edition.

Unless otherwise specified, any statement of optional behavior in this specification that is prescribed using the terms SHOULD or SHOULD NOT implies product behavior in accordance with the SHOULD or SHOULD NOT prescription. Unless otherwise specified, the term MAY implies that the product does not follow the prescription.

<1> Section 1.7: The ICertPassage Interface is supported by all applicable Windows products. However, CMC (Certificate Management Protocol using CMS) request formats and CMC response formats are not supported by Windows NT or Windows 2000. CMC is specified in [RFC2797].

<2> Section 2.2.1: Windows NT and Windows 2000 do not support the CMC request format (as specified in [RFC2797]).

<3> Section 3.1.3: For authenticated RPC, the client in Windows XP and all subsequent versions of Windows, according to the applicability list at the beginning of this section, passes RPC_C_AUTHN_GSS_NEGOTIATE and RPC_C_AUTHN_LEVEL_PKT_PRIVACY. The client in Windows 2000 Server operating system passes RPC_C_AUTHN_GSS_NEGOTIATE only to RPC. These values are used to allow RPC to negotiate the authentication level on behalf of the client with the server, as specified in [MS-RPCE].

<4> Section 3.2.3: The exchange private key was not used prior to Windows Server 2003.

<5> Section 3.2.4.1: The supported clients are:

- Windows 2000 Professional operating system
- Windows XP
- Windows Vista
- Windows 7
- Windows 8
- Windows 8.1
- Windows 10

The supported servers are:

- Windows NT Server operating system
- Windows 2000 Server
- Windows Server 2003
- Windows Server 2008
- Windows Server 2008 R2
- Windows Server 2012
- Windows Server 2012 R2
- Windows Server 2016~~ Technical Preview~~

<6> Section 3.2.4.1.1: The implementation of this method on Windows Server 2003 and all subsequent versions of Windows Server operating system, according to the applicability list at the beginning of this section, is identical to the ICertRequestD::Request method, as specified in [MS-WCCE] section 3.2.1.4.2.1. However, the implementation of this method on Windows 2000 Server has the following differences from the ICertRequestD::Request method. In ICertPassage on Windows 2000 Server:

- The format of the certificates request passed in the *pctbRequest* parameter ~~MUST NOT~~cannot be CMC, as specified in [RFC2797].
- Windows 2000 Server does not return or support issued certificates in the CMC format, as specified in [RFC2797].

<7> Section 3.2.4.1.1: In Windows Server 2003 and all subsequent versions of Windows Server, according to the applicability list at the beginning of this section, the error returned is RPC_S_SERVER_UNAVAILABLE (0x800706ba). Windows 2000 does not return an error.

8 Change Tracking

~~No table of~~ This section identifies changes ~~is available. The~~that were made to this document ~~is either new or has had no changes~~ since ~~its~~the last release. Changes are classified as New, Major, Minor, Editorial, or No change.

The revision class **New** means that a new document is being released.

The revision class **Major** means that the technical content in the document was significantly revised. Major changes affect protocol interoperability or implementation. Examples of major changes are:

- A document revision that incorporates changes to interoperability requirements or functionality.
- The removal of a document from the documentation set.

The revision class **Minor** means that the meaning of the technical content was clarified. Minor changes do not affect protocol interoperability or implementation. Examples of minor changes are updates to clarify ambiguity at the sentence, paragraph, or table level.

The revision class **Editorial** means that the formatting in the technical content was changed. Editorial changes apply to grammatical, formatting, and style issues.

The revision class **No change** means that no new technical changes were introduced. Minor editorial and formatting changes may have been made, but the technical content of the document is identical to the last released version.

Major and minor changes can be described further using the following change types:

- New content added.
- Content updated.
- Content removed.
- New product behavior note added.
- Product behavior note updated.
- Product behavior note removed.
- New protocol syntax added.
- Protocol syntax updated.
- Protocol syntax removed.
- New content added due to protocol revision.
- Content updated due to protocol revision.
- Content removed due to protocol revision.
- New protocol syntax added due to protocol revision.
- Protocol syntax updated due to protocol revision.
- Protocol syntax removed due to protocol revision.
- Obsolete document removed.

Editorial changes are always classified with the change type **Editorially updated**.

Some important terms used in the change type descriptions are defined as follows:

- **Protocol syntax** refers to data elements (such as packets, structures, enumerations, and methods) as well as interfaces.
- **Protocol revision** refers to changes made to a protocol that affect the bits that are sent over the wire.

The changes made to this document are listed in the following table. For more information, please contact dochelp@microsoft.com.

Section	Tracking number (if applicable) and description	Major change (Y or N)	Change type
2.2 Common Data Types	Removed mention of the fact that the protocol makes use of the BYTE datatype.	Y	Content update.
3.2.4.1.1 CertServerRequest (Opnum 0)	Changed the datatype of the h parameter from const handle_t to handle_t to match the IDL.	Y	Content update.
6 Appendix A: Full IDL	Removed the definition of BYTE.	Y	Content update.

9 Index

L

M

N

O

P

R

S

T

V