[MS-ICPR]:
ICertPassage Remote Protocol Specification

Revision Summary

Date	Revision History	Revision Class	Comments
05/11/2007	0.1		MCPP Milestone 4 Initial Availability
08/10/2007	0.1.1	Editorial	Revised and edited the technical content.
09/28/2007	0.1.2	Editorial	Revised and edited the technical content.
10/23/2007	0.1.3	Editorial	Revised and edited the technical content.
11/30/2007	0.1.4	Editorial	Revised and edited the technical content.

Date	Revision History	Revision Class	Comments
01/25/2008	0.1.5	Editorial	Revised and edited 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.

1.1 Glossary

The following terms are defined in [MS-GLOS]:

Active Directory (AD)
Certificate
Certificate Authority (CA) (or Certification Authority)
Certification
Client
Digital Signature
Distributed Component Object Model (DCOM)
Endpoint
Endpoint Mapper
Enroll
Enrollment
Private Key
Public Key
Public-Private Key Pair
Remote Procedure Call (RPC)
RPC Endpoint
Universally Unique Identifier (UUID)
Well-Known Endpoint

The following terms are specific to this document:

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

1.2 References

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. Please check the archive site, http://msdn2.microsoft.com/en-us/library/E4BD6494-06AD-4aed-9823-445E921C9624, as an additional source.

[IEEE1363] Institute of Electrical and Electronics Engineers, "Standard Specifications for Public-Key Cryptography", 1363-2000, August 1999, http://grouper.ieee.org/groups/1363/

[MS-CRTD] Microsoft Corporation, "Certificate Templates Structure Specification", March 2007.

[MS-DCOM] Microsoft Corporation, "Distributed Component Object Model (DCOM) Remote Protocol Specification", March 2007.

[MS-RPCE] Microsoft Corporation, "Remote Procedure Call Protocol Extensions", January 2007.

[MS-WCCE] Microsoft Corporation, "Windows Client Certificate Enrollment Protocol Specification", June 2007.

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

Note There is a charge to download the specification.

[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

Note There is a charge to download the specification.

[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

Note There is a charge to download the specification.

1.2.2 Informative References

[CRYPTO] Menezes, A., Vanstone, S., and Oorschot, P., "Handbook of Applied Cryptography", 1997, http://www.cacr.math.uwaterloo.ca/hac/

[SCHNEIER] Schneier, B., "Applied Cryptography, Second Edition", John Wiley and Sons, 1996, ISBN: 0471117099.

If you have any trouble finding [SCHNEIER], please check here.

1.3 Protocol Overview (Synopsis)

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 3.1.1)**.<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.

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.

1.6 Applicability Statement

This protocol applies to legacy clients that must 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 over a **Distributed Component Object Model (DCOM)** (as specified in [MS-DCOM]), they should 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.<2>

1.8 Vendor-Extensible Fields

This protocol contains no vendor-extensible fields.

1.9 Standards Assignments

There are no standards assigned to this protocol.

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 this protocol, using RPC over TCP, the TCP endpoint is dynamic. For more information, see Client Initialization (section 4.1.3) and Server Initialization (section 4.2.3).

This protocol MUST use the **universal unique identifier (UUID)**, as specified in section 3.1.

2.2 Message Syntax

2.2.1 Common Structures

The ICertPassage interface uses the CERTTRANSBLOB structure, as specified in [MS-WCCE] section **2.2.1.1**.

2.2.1.1 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].
- PKCS #7 as specified in [RFC3852].
- CMC as specified in [RFC2797].

Details are as specified in [MS-WCCE] section 2.2.1.4. Each format contains a set of attributes and extensions describing the request.<3>

2.2.1.2 Response Format

Responses are returned by the ICertPassage Remote Protocol in either PKCS #7 format or CMC format. Details are as specified in [MS-WCCE] section 2.2.1.6. The format of the response is determined by the value passed in the *dwFlags* parameter, as specified in section 4.1.5.1.

3 ICertPassage Interface

The ICertPassage Remote Protocol defines the following interface:

ICertPassage (section 3.1): A method that enables a client to request certificates from a certification authority.

3.1 ICertPassage

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 1.0. The UUID for this interface is 91ae6020-9e3c-11cf-8d7c-00aa00c091be, as specified in [MS-RPCE].<4>

The interface defines a single method:

Methods in RPC Opnum Order

Method	Description
CertServerRequest	Opnum: 0

3.1.1 CertServerRequest (Opnum 0)

The **CertServerRequest** method processes a certificate enrollment request from the client.<5>

```
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
);
```

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, as specified in [MS-WCCE] section **3.2.4.1.1.1**.

pwszAuthority: The *pwszAuthority* parameter has identical syntax and semantics to the *pwszAuthority* parameter, as specified in [MS-WCCE] section **3.2.4.1.1.1**.

pdwRequestId: The *pdwRequestId* parameter has identical syntax and semantics to the *pdwRequestId* parameter, as specified in [MS-WCCE] section **3.2.4.1.1.1**.

pdwDisposition: The *pdwDisposition* parameter has identical syntax and semantics to the *pdwDisposition* parameter, as specified in [MS-WCCE] section **3.2.4.1.1.1**.

pctbAttribs: A pointer to a **CERTTRANSBLOB** structure, as specified in [MS-WCCE] section **2.2.1.1**, where the *pb* parameter points to a Unicode (as specified in [UNICODE4.0]) NULL-terminated string and the *cb* parameter contains the length of the string, including the NULL-terminated character (in bytes). The semantics of the string are identical to the *pwszAttributes* parameter, as specified in [MS-WCCE] section **3.2.4.1.1.1**.

pctbRequest: The *pctbRequest* parameter has identical syntax and semantics to the *pctbRequest* parameter, as specified in [MS-WCCE] section **3.2.4.1.1.1**.

pctbCert: The *pctbCert* parameter has identical syntax and semantics to the *pctbCertChain* parameter, as specified in [MS-WCCE] section **3.2.4.1.1.1**.

pctbEncodedCert: The *pctbEncodedCert* parameter has identical syntax and semantics to the *pctbEncodedCert* parameter, as specified in [MS-WCCE] section **3.2.4.1.1.1**.

pctbDispositionMessage: The *pctbDispositionMessage* parameter has identical syntax and semantics to the *pctbDispositionMessage* parameter, as specified in [MS-WCCE] section **3.2.4.1.1.1**.

Return Values: The method MUST return ERROR_SUCCESS (0x00000000) on success and SHOULD return additional processing status in the methods parameters *pdwDisposition* and *pctbDispositionMessage*. Otherwise, the CA MUST return a nonzero value; the value returned SHOULD be one of the values specified in [MS-ERREF].

4 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. In the other cases, a human administrative decision is required by the CA before it will issue the requested certificate. The server indicates this situation to the client by returning a value of CR_DISP_UNDER_SUBMISSION (0x00000005) in the *pdwDisposition* parameter of the **CertServerRequest** method. Details on the flow and sequencing of the certificate enrollment protocol are as specified in [MS-WCCE] sections 3.1.4 and 3.2.4.

4.1 Client Role

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

4.1.1 Abstract Data Model

The client abstract data model is exactly as specified in [MS-WCCE] section 3.1.1.

4.1.2 Timers

There are no timers in this protocol.

4.1.3 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 MAY 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]. 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.<6>

4.1.4 Higher-Layer Triggered Events

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

4.1.5 Message Processing and Sequencing Rules

4.1.5.1 Processing ICertPassage:: CertServerRequest

Details of the client processing rules are exactly as specified in [MS-WCCE] section **3.1.4.4**.

4.2 Server Role

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

4.2.1 Abstract Data Model

As specified in [MS-WCCE] section 3.2.1.

4.2.2 Timers

There are no timers in this protocol.

4.2.3 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.<7>

4.2.4 Message Processing and Sequencing Rules

As specified in [MS-WCCE] section 3.2.4.

4.2.4.1 Processing ICertPassage::CertServerRequest

See the processing rules for the ICertRequestD::Request method, as specified in [MS-WCCE] section 3.1.4.5.

5 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 (AD)** for the templates (as specified in [MS-CRTD]) that are available for the specified user. As the resource manager, AD enforces that the user only receives templates that the user has read permissions to access (as specified in [MS-WCCE] section 3.1.4.2).

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 PKCS#7 request message (as specified in [MS-WCCE] section 2.2.1.4.2), 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 4.1.3.

5. Using the connection above, the client invokes the **ICertPassage::CertServerRequest()** method (see section 4.2.4.1), and, in doing this, 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.

6 Security Considerations

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

7 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);
}
```

8 Appendix B: Windows Behavior

The information in this specification is applicable to the following versions of Windows:

- Windows NT
- Windows 2000
- Windows XP
- Windows Server 2003
- Windows Vista
- Windows Server 2008

Exceptions, if any, are noted below. Unless otherwise specified, any statement of optional behavior in this specification prescribed using the terms SHOULD or SHOULD NOT implies Windows behavior in accordance with the SHOULD or SHOULD NOT prescription. Unless otherwise specified, the term MAY implies that Windows does not follow the prescription.

<1> Section 1.3: The ICertPassage Remote Protocol was first introduced in Windows NT 4.0 as the first certificate enrollment protocol for the Windows operating system. The Windows Client Certificate Enrollment Protocol, as specified in [MS-WCCE], was introduced in Windows 2000 Server as the preferred certificate enrollment protocol and as a replacement for the ICertPassage Remote Protocol.

<2> Section 1.7: The interface is supported by all versions of Windows, beginning with Windows NT 4.0. However, CMC (Certificate Management Protocol using CMS) request formats and CMC response formats are supported only by Windows XP, Windows Server 2003, Windows Vista, and Windows Server 2008. CMC is specified in [RFC2797].

<3> Section 2.2.1.1: Only Windows XP, Windows Server 2003, Windows Vista, and Windows Server 2008 support the CMC request format (as specified in [RFC2797]).

<4> Section 3.1: The supported clients are Windows Vista, Windows XP Professional, Windows XP Home Edition, Windows 2000 Professional, Windows NT 4.0, Windows Server 2008, Windows Server 2003, and Windows 2000 Server. The supported servers are Windows Server 2008, Windows Server 2003, Windows 2000 Server, and Windows NT Server.

<5> Section 3.1.1: The implementation of this method on Windows Server 2003 is identical to the ICertRequestD::Request method, as specified in [MS-WCCE] section **3.2.4.1.1.1**. 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 be CMC, as specified in [RFC2797].
- Windows 2000 Server does not return or support issued certificates in the CMC format, as specified in [RFC2797].

<6> Section 4.1.3: Windows-based clients create a separate binding for every method invocation. For authenticated RPC, the client in Windows XP and Windows Server 2003 passes RPC_C_AUTHN_GSS_NEGOTIATE and RPC_C_AUTHN_LEVEL_PKT_PRIVACY. The client in Windows 2000 Server 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].

<7> Section 4.2.3: The exchange private key was not used prior to Windows Server 2003.

9 Index

T

V

W