

[MS-SSP]: Single Sign-On Protocol Specification

Intellectual Property Rights Notice for Open Specifications Documentation

- **Technical Documentation.** Microsoft publishes Open Specifications documentation for protocols, file formats, languages, standards as well as overviews of the interaction among each of these technologies.
- **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 make copies of it in order to develop implementations of the technologies described in the Open Specifications and may distribute portions of it in your implementations using these technologies or your documentation as necessary to properly document the implementation. You may also distribute in your implementation, with or without modification, any schema, IDL's, or code samples that are included in the documentation. This permission also applies to any documents that are referenced in the Open Specifications.
- **No Trade Secrets.** Microsoft does not claim any trade secret rights in this documentation.
- **Patents.** Microsoft has patents that may cover your implementations of the technologies described in the Open Specifications. Neither this notice nor Microsoft's delivery of the documentation grants any licenses under those or any other Microsoft patents. However, a given Open Specification may be covered by Microsoft [Open Specification Promise](#) or the [Community Promise](#). If you would prefer a written license, or if the technologies described in the Open Specifications 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 be covered by trademarks or similar intellectual property rights. This notice does not grant any licenses under those rights.
- **Fictitious Names.** The example companies, organizations, products, domain names, e-mail addresses, logos, people, places, and events 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 specifically described above, whether by implication, estoppel, or otherwise.

Tools. The Open Specifications do 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 are intended for use in conjunction with publicly available standard specifications and network programming art, and assumes that the reader either is familiar with the aforementioned material or has immediate access to it.

Revision Summary

Date	Revision History	Revision Class	Comments
04/04/2008	0.1		Initial Availability
06/27/2008	1.0	Major	Revised and edited the technical content
12/12/2008	1.01	Editorial	Revised and edited the technical content
07/13/2009	1.02	Major	Changes made for template compliance
08/28/2009	1.03	Editorial	Revised and edited the technical content
11/06/2009	1.04	Editorial	Revised and edited the technical content
02/19/2010	2.0	Major	Updated and revised the technical content
03/31/2010	2.01	Editorial	Revised and edited the technical content
04/30/2010	2.02	Minor	Updated the technical content
06/07/2010	2.03	Editorial	Revised and edited the technical content
06/29/2010	2.04	Editorial	Changed language and formatting in the technical content.
07/23/2010	2.04	No change	No changes to the meaning, language, or formatting of the technical content.
09/27/2010	2.04	No change	No changes to the meaning, language, or formatting of the technical content.
11/15/2010	2.04	No change	No changes to the meaning, language, or formatting of the technical content.
12/17/2010	2.04	No change	No changes to the meaning, language, or formatting of the technical content.
03/18/2011	2.04	No change	No changes to the meaning, language, or formatting of the technical content.
06/10/2011	2.04	No change	No changes to the meaning, language, or formatting of the technical content.
01/20/2012	2.04	No change	No changes to the meaning, language, or formatting of the technical content.
04/11/2012	2.04	No change	No changes to the meaning, language, or formatting of the technical content.
07/16/2012	2.04	No change	No changes to the meaning, language, or formatting of the technical content.
09/12/2012	2.04	No change	No changes to the meaning, language, or formatting of the technical content.

Date	Revision History	Revision Class	Comments
10/08/2012	2.5	Minor	Clarified the meaning of the technical content.

Table of Contents

1 Introduction	5
1.1 Glossary	5
1.2 References	5
1.2.1 Normative References	5
1.2.2 Informative References	6
1.3 Overview	6
1.4 Relationship to Other Protocols	6
1.5 Prerequisites/Preconditions	6
1.6 Applicability Statement	6
1.7 Versioning and Capability Negotiation	6
1.8 Vendor-Extensible Fields	6
1.9 Standards Assignments	6
2 Messages	7
2.1 Transport	7
2.2 Common Data Types	7
2.2.1 MAX_MASTER_SECRET_BYTES	7
2.2.2 SizeOfMasterSecretInBytes	7
2.2.3 error_status_t	7
3 Protocol Details	8
3.1 Protocol Server Details	8
3.1.1 Abstract Data Model	8
3.1.2 Timers	8
3.1.3 Initialization	8
3.1.4 Message Processing Events and Sequencing Rules	8
3.1.4.1 RemoteGetMasterSecret (Opnum 0)	8
3.1.5 Timer Events	9
3.1.6 Other Local Events	10
3.2 Protocol Client Details	10
3.2.1 Abstract Data Model	10
3.2.2 Timers	10
3.2.3 Initialization	10
3.2.4 Message Processing Events and Sequencing Rules	10
3.2.5 Timer Events	10
3.2.6 Other Local Events	10
4 Protocol Examples	11
5 Security	12
5.1 Security Considerations for Implementers	12
5.2 Index of Security Parameters	12
6 Appendix A: Full IDL	13
7 Appendix B: Product Behavior	14
8 Change Tracking	15
9 Index	17

1 Introduction

This document specifies the Single Sign-On Protocol, which protocol clients use to obtain the key that is used to symmetrically encrypt and decrypt credentials and single sign-on (SSO) tickets from the protocol server.

Sections 1.8, 2, and 3 of this specification are normative and can contain the terms MAY, SHOULD, MUST, MUST NOT, and SHOULD NOT as defined in RFC 2119. Sections 1.5 and 1.9 are also normative but cannot contain those terms. All other sections and examples in this specification are informative.

1.1 Glossary

The following terms are defined in [\[MS-GLOS\]](#):

dynamic endpoint
IDL
opnum
remote procedure call (RPC)
RPC transport
universally unique identifier (UUID)
user object

The following terms are defined in [\[MS-OFCGLOS\]](#):

master secret
master secret server
security principal

The following terms are specific to this document:

single sign-on (SSO) administrator: A security principal (2) who is authorized to change a single sign-on (SSO) configuration and to obtain master secrets from a master secret server.

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

References to Microsoft Open Specifications documentation do not include a publishing year because links are to the latest version of the technical documents, which are updated frequently. References to other documents include a publishing year when one is available.

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.

[C706] The Open Group, "DCE 1.1: Remote Procedure Call", C706, August 1997, <http://www.opengroup.org/public/pubs/catalog/c706.htm>

[MS-ERREF] Microsoft Corporation, "[Windows Error Codes](#)".

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

[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>

1.2.2 Informative References

[MS-GLOS] Microsoft Corporation, "[Windows Protocols Master Glossary](#)".

[MS-OFCGLOS] Microsoft Corporation, "[Microsoft Office Master Glossary](#)".

1.3 Overview

The Single Sign-On Service protocol is a **remote procedure call (RPC)**-based protocol used by protocol clients to obtain a **master secret** from a **master secret server**.

1.4 Relationship to Other Protocols

This protocol depends on remote procedure call (RPC) for its transport.

This protocol uses RPC over TCP/IP as specified in section [2.1](#).

1.5 Prerequisites/Preconditions

The protocol is an RPC interface and therefore uses the common RPC interface prerequisites as described in [\[MS-RPCE\]](#).

The protocol requires the protocol client to obtain the name of the remote computer that supports this protocol before the protocol client calls methods that are associated with the remote computer. How a protocol client accomplishes this is outside of the scope of this specification.

1.6 Applicability Statement

This protocol is designed to be used only by **single sign-on (SSO) administrator security principals (2)**. The information associated with this protocol can contain highly sensitive information, so the protocol client is designed to be used in an environment that is appropriately secured.

1.7 Versioning and Capability Negotiation

None.

1.8 Vendor-Extensible Fields

This protocol uses Win32 error codes, as specified in [\[MS-ERREF\]](#) section 2.2. Vendors SHOULD [<1>](#) reuse those values with their indicated meanings. Choosing any other value runs the risk of a collision in the future.

1.9 Standards Assignments

Parameter	Value	Reference
RPC interface UUID	9D07CA0D-8F02-4ed5-B727-ACF37FEA5BBC	[C706]

2 Messages

2.1 Transport

This protocol uses RPC **dynamic endpoints** as specified in [C706], part 4. This protocol allows any **user object** to establish a connection to the RPC server. The protocol server uses RPC as the underlying protocol to retrieve the identity of the caller that made the method call as specified in [MS-RPCE] section 3.3.3.4.3. The protocol server SHOULD<2> use this identity to perform method-specific permission checks as specified in section 3.

2.2 Common Data Types

This section specifies data types, in addition to RPC base types and definitions specified in [C706] and [MS-RPCE].

The following table summarizes the common data types used by this specification.

Data type	Description
MAX_MASTER_SECRET_BYTES (section 2.2.1)	The maximum number of bytes that the master secret can contain.
SizeOfMasterSecretInBytes (section 2.2.2)	The size in bytes of the master secret.
error_status_t (section 2.2.3)	The error code number reported by this protocol.

2.2.1 MAX_MASTER_SECRET_BYTES

The maximum number of bytes that the master secret can contain.

```
const unsigned long MAX_MASTER_SECRET_BYTES = 16;
```

2.2.2 SizeOfMasterSecretInBytes

The size of the master secret, specified in bytes. It MUST be in a range having 0 as the low allowed value and **MAX_MASTER_SECRET_BYTES** as the high allowed value.

```
typedef [range(MAX_MASTER_SECRET_BYTES,MAX_MASTER_SECRET_BYTES)] unsigned long  
SizeOfMasterSecretInBytes;
```

2.2.3 error_status_t

The error code number reported by the protocol.

```
typedef unsigned long error_status_t;
```

This protocol MUST configure the RPC runtime to serialize this data type as specified in [MS-RPCE] section 2.1.

3 Protocol Details

The protocol client is simply a pass-through. That is, no additional timers or other state is required by the protocol client. Calls made by the higher-layer protocol or application are passed directly to the transport and the results returned by the transport are passed directly back to the higher-layer protocol or application.

The protocol method behaves in the same manner, regardless of whether the protocol server or the protocol client is running in a client or server version of the operating system. The protocol method returns the value "0x00000000" if it finishes processing successfully. If it fails, it sends a nonzero, implementation-specific error code. The protocol server uses nonzero Win32 error values to signify error conditions, as specified in section [1.8](#), unless the application requires other implementation-specific values.

The protocol client MUST NOT take action on the error codes it receives. It MUST send them to the calling application.

3.1 Protocol Server Details

3.1.1 Abstract Data Model

This section describes a conceptual model of possible data organization that an implementation maintains to participate in this protocol. The described organization is provided to facilitate the explanation of how the protocol behaves. This document does not mandate that implementations adhere to this model as long as their external behavior is consistent with that described in this document.

A protocol server that implements this RPC interface MUST verify that the protocol client user object is an single sign-on (SSO) administrator and that the protocol server can retrieve the master secret.

3.1.2 Timers

None.

3.1.3 Initialization

Parameters necessary to initialize the RPC protocol are specified in section [2.1](#).

3.1.4 Message Processing Events and Sequencing Rules

The protocol server SHOULD [enforce](#) security measures to verify that the user object was granted the permissions required to request the following routines.

This interface includes the following method:

Method	Description
RemoteGetMasterSecret	Returns the master secret. opnum : 0

3.1.4.1 RemoteGetMasterSecret (Opnum 0)

The **RemoteGetMasterSecret** method returns the master secret. A protocol server that implements this RPC interface MUST verify that the protocol client user object is a single sign-on

(SSO) administrator before it retrieves the master secret. Calls associated with any other user object MUST fail.

```
error_status_t
RemoteGetMasterSecret(
[in] handle_t hBinding,
[in, out] SizeOfMasterSecretInBytes * pcbSecret,
[size_is(*pcbSecret), out] byte * pbSecret);
```

hBinding: Specifies the pointer to the protocol client remote procedure call (RPC) binding handle. The **handle_t** data type represents an explicit remote procedure call (RPC) binding handle, as specified in [\[C706\]](#).

pcbSecret: On input, this parameter points to a value of type **SizeOfMasterSecretInBytes** as specified in section [2.2.2](#). On output, this parameter MUST specify the actual number of bytes that the master secret contains.

pbSecret: A pointer of type **SizeOfMasterSecretInBytes** that points to the master secret.

Return CodeValues: An unsigned long integer whose values are specified in the following table.

Value	Description
ERROR_SUCCESS	Contains the value "0x00000000" and specifies that no error encountered.
SSO_E_ACCESSDENIED	Contains the value "0x80630005" and specifies that the user object is not a single sign-on (SSO) administrator.
E_OUTOFMEMORY	Contains the value "0x8007000E" and specifies that the input value of the <i>pcbSecret</i> parameter is less than number of bytes needed for the master secret.
E_INVALIDARG	Contains the value "0x80070057" and specifies that the method parameters are NULL or the <i>pcbSecret</i> parameter points to a value that is less than or equal to 0.
E_UNEXPECTED	Contains the value "0x8000FFFF" and specifies that the protocol server that processes the call is not properly initialized.
RPC_S_SERVER_UNAVAILABLE	Contains the value "0xC0020017" and specifies that the protocol server that processes the call it's shutting down.
SSO_E_EXCEPTION	Contains the value "0x80630428" and specifies that the protocol server that processes the call threw an exception.
SSO_E_MASTER_SECRET_NOT_EXIST	Contains the value "0x80630002" and specifies that the master secret does not exist.
SSO_E_SSO_NOT_CONFIGURED	Contains the value "0x8063064A" and specifies that the protocol server that processes the call is not configured.

Exceptions Thrown: No exceptions are thrown other than those thrown by the underlying protocol.

3.1.5 Timer Events

None.

3.1.6 Other Local Events

There are no local events used on the protocol server other than the events maintained in the underlying **RPC transport**.

3.2 Protocol Client Details

3.2.1 Abstract Data Model

None.

3.2.2 Timers

None.

3.2.3 Initialization

The protocol client MUST create an RPC connection to the remote computer, using the information specified in section [2.1](#).

3.2.4 Message Processing Events and Sequencing Rules

This is a stateless protocol. No sequence of method calls is imposed on this protocol. When a method finishes processing, the values returned by the underlying protocol server MUST be returned unmodified to the upper layer by the protocol client.

The protocol client MUST ignore errors received from the protocol server and notify the calling application of the errors it received. It MUST also interact with the underlying RPC protocol as specified in [\[MS-RPCE\]](#) for message processing. Otherwise, the protocol client does not perform any special message processing.

3.2.5 Timer Events

None.

3.2.6 Other Local Events

None.

4 Protocol Examples

In this scenario, the protocol client calls the **RemoteGetMasterSecret** method on a protocol server whose URL contains "server.example.com". The protocol client originates the binding information that the protocol server uses to authenticate the protocol client. The protocol client allocates **MAX_MASTER_SECRET_BYTES** bytes for the *pbSecret* parameter before calling the remote method.

```
RemoteGetMasterSecret([in] handle_t hBinding,  
[in, out] unsigned long * pcbSecret,  
[out] BYTE * pbSecret);
```

On receiving this request the protocol server executes the method locally and returns:

```
Return value is: ERROR_SUCCESS(an error_status_t value)  
RemoteGetMasterSecret([in] handle_t hBinding,  
[in, out] unsigned long * pcbSecret,  
[ out] BYTE * pbSecret);
```

5 Security

5.1 Security Considerations for Implementers

This protocol allows any user object to connect to the protocol server, as specified in section [2.1](#). The implementation enforces security on the method to reduce exploitable security issues. If the user object is not a single sign-on (SSO) administrator, the protocol fails as specified in section [3.1.4.1](#).

5.2 Index of Security Parameters

Security Parameter	Section
Authentication Protocol	2.1

6 Appendix A: Full IDL

For ease of implementation, the full **IDL** is provided in this section, where "ms-rpce.idl" refers to the IDL found in [MS-RPCE] Appendix A. The syntax uses the IDL syntax extensions defined in [MS-RPCE] Sections 2.2.4 and 3.1.1.5. For example, as noted in [MS-RPCE] section 2.2.4.9, a `pointer_default` declaration is not required and `pointer_default(unique)` is assumed.

```
import "ms-rpce.idl";
const unsigned long MAX_MASTER_SECRET_BYTES=16;
typedef [range(MAX_MASTER_SECRET_BYTES,MAX_MASTER_SECRET_BYTES)] unsigned long
SizeOfMasterSecretInBytes;
[
  uuid(9D07CA0D-8F02-4ed5-B727-ACF37FEA5BBC),
  version(1.0),
  pointer_default(unique)
]
interface ISingleSignonRemoteMasterSecret
{
  error_status_t RemoteGetMasterSecret(
  [in] handle_t hBinding,
  [in, out] SizeOfMasterSecretInBytes * pcbSecret,
  [size_is(*pcbSecret),
  out] byte * pbSecret);
}
```

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:

- Microsoft® Office SharePoint® Portal Server 2003
- Microsoft® Office SharePoint® Server 2007
- Microsoft® SQL Server® 2005
- Microsoft® SQL Server® 2008
- Microsoft® SQL Server® 2008 R2
- Microsoft® SQL Server® 2008 R2 SP1
- Microsoft® SQL Server® 2012
- Microsoft® SharePoint® Server 2010
- Microsoft® SharePoint® Server 2013

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 1.8:](#) Windows uses only the values that are described in [\[MS-ERREF\]](#) section 2.2.

[<2> Section 2.1:](#) Windows uses the identity of the caller to perform method specific access checks.

[<3> Section 3.1.4:](#) Office SharePoint Server 2007 uses the underlying Windows security subsystem to determine the permissions for the caller.

8 Change Tracking

This section identifies changes that were made to the [MS-SSP] protocol document between the September 2012 and October 2012 releases. 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.
- An extensive rewrite, addition, or deletion of major portions of content.
- The removal of a document from the documentation set.
- Changes made for template compliance.

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 language and 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 or language changes were introduced. The technical content of the document is identical to the last released version, but minor editorial and formatting changes, as well as updates to the header and footer information, and to the revision summary, may have been made.

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.
- New content added for template compliance.
- Content updated for template compliance.
- Content removed for template compliance.
- 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 protocol@microsoft.com.

Section	Tracking number (if applicable) and description	Major change (Y or N)	Change type
1.8 Vendor-Extensible Fields	Updated the description for clarity.	N	Content removed.
Z Appendix B: Product Behavior	Updated product name list.	N	New content added.

9 Index

A

Abstract data model
[client](#) 10
[server](#) 8
[Applicability](#) 6

C

[Capability negotiation](#) 6
[Change tracking](#) 15
Client
[abstract data model](#) 10
[initialization](#) 10
[local events](#) 10
[message processing](#) 10
[overview](#) 8
[sequencing rules](#) 10
[timer events](#) 10
[timers](#) 10
[Common data types](#) 7

D

Data model - abstract
[client](#) 10
[server](#) 8
Data types
[common - overview](#) 7
[error_status_t](#) 7
[MAX_MASTER_SECRET_BYTES](#) 7
[SizeOfMasterSecretInBytes](#) 7

E

[error_status_t data type](#) 7
Events
[local - client](#) 10
[local - server](#) 10
[timer - client](#) 10
[timer - server](#) 9
Examples
[overview](#) 11

F

[Fields - vendor-extensible](#) 6
[Full IDL](#) 13

G

[Glossary](#) 5

I

[IDL](#) 13
[Implementer - security considerations](#) 12
[Index of security parameters](#) 12

[Informative references](#) 6

Initialization
[client](#) 10
[server](#) 8
[Introduction](#) 5

L

Local events
[client](#) 10
[server](#) 10

M

[MAX_MASTER_SECRET_BYTES data type](#) 7
Message processing
[client](#) 10
[server](#) 8
Messages
[common data types](#) 7
[error_status_t data type](#) 7
[MAX_MASTER_SECRET_BYTES data type](#) 7
[SizeOfMasterSecretInBytes data type](#) 7
[transport](#) 7
Methods
[RemoteGetMasterSecret \(Opnum 0\)](#) 8

N

[Normative references](#) 5

O

[Overview \(synopsis\)](#) 6

P

[Parameters - security index](#) 12
[Preconditions](#) 6
[Prerequisites](#) 6
[Product behavior](#) 14

R

[References](#) 5
[informative](#) 6
[normative](#) 5
[Relationship to other protocols](#) 6
[RemoteGetMasterSecret \(Opnum 0\) method](#) 8

S

Security
[implementer considerations](#) 12
[parameter index](#) 12
Sequencing rules
[client](#) 10
[server](#) 8

Server

- [abstract data model](#) 8
- [initialization](#) 8
- [local events](#) 10
- [message processing](#) 8
- [overview](#) 8
- [RemoteGetMasterSecret \(Opnum 0\) method](#) 8
- [sequencing rules](#) 8
- [timer events](#) 9
- [timers](#) 8
- [SizeOfMasterSecretInBytes data type](#) 7
- [Standards assignments](#) 6

T

Timer events

- [client](#) 10
- [server](#) 9

Timers

- [client](#) 10
- [server](#) 8
- [Tracking changes](#) 15
- [Transport](#) 7

V

- [Vendor-extensible fields](#) 6
- [Versioning](#) 6