[MS-SNID]:
Server Network Information Discovery Protocol

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. For a list of Microsoft trademarks, visit www.microsoft.com/trademarks.
- **Fictitious Names.** The example companies, organizations, products, domain names, email 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
08/08/2013	1.0	New	Released new document.
11/14/2013	2.0	Major	Significantly changed the technical content.

Contents

1 Introduction

The Server Network Information Discovery Protocol enables protocol clients to discover protocol servers within a UDP broadcast boundary and get server's networking configuration information.

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]:

Domain Name System (DNS)
Internet Protocol version 4 (IPv4)
Internet Protocol version 6 (IPv6)
NetBIOS name
User Datagram Protocol (UDP)

1.2 References

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

A reference marked "(Archived)" means that the reference document was either retired and is no longer being maintained or was replaced with a new document that provides current implementation details. We archive our documents online [Windows Protocol].

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.

[RFC768] Postel, J., "User Datagram Protocol", STD 6, RFC 768, August 1980, http://www.ietf.org/rfc/rfc768.txt

[RFC919] Mogul, J., "BROADCASTING INTERNET DATAGRAMS", RFC 919, October 1984, http://www.rfc-editor.org/rfc/rfc919.txt

1.2.2 Informative References

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

[RFC2375] Hinden, R., and Deering, S., "IPv6 Multicast Address Assignments", RFC 2375, July 1998, http://www.rfc-editor.org/rfc/rfc2375.txt

1.3 Overview

The Server Network Information Discovery Protocol defines a pair of request and response messages by which a protocol client can locate protocol servers within the broadcast/multicast scope and get network information (such as **NetBIOS name**, **Internet Protocol version 4 (IPv4)**, and **Internet Protocol version 6 (IPv6)** addresses) of the servers.

1.4 Relationship to Other Protocols

The protocol relies on **User Datagram Protocol (UDP)** as specified in [RFC 768].

1.5 Prerequisites/Preconditions

The protocol server allows the incoming UDP package from port 8912 in the firewall.

1.6 Applicability Statement

The protocol server and client are connected to the same subnet with IPv4 broadcast support or IPv4 link-local scope multicast support.

1.7 Versioning and Capability Negotiation

The protocol version specifies the interoperability capability of the protocol on different Windows operating systems. See section 6 for detailed product behaviors of protocol versions and corresponding supported operating system. Protocol servers and clients with different protocol versions are not able to negotiate with each other.

1.8 Vendor-Extensible Fields

None.

1.9 Standards Assignments

Parameter	Value	Reference
UDP port	8912	http://www.iana.org/assignments/service-names-port-numbers/service-names-port-numbers.xml

2 Messages

2.1 Transport

The protocol transports messages over UDP as specified in [RFC768]. A protocol client MUST send one UDP package to the broadcast (IPv4) address and multicast (IPv6 link local scope all nodes) address using UDP port 8912. Protocol servers MUST reply to the protocol client with a UDP package containing its NetBIOS name and **DNS** configuration of the protocol server as specified later in this section.

2.2 Message Syntax

2.2.1 Enumerations

2.2.1.1 Id Enumeration

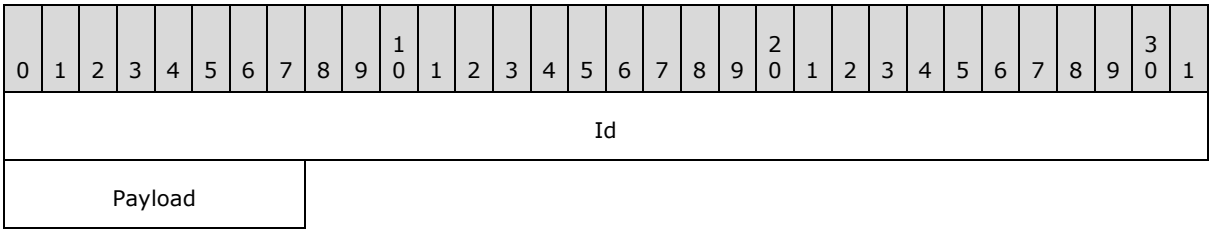
The **Id** enumeration is used to distinguish the Network Information Discovery request and Network Information Discovery response package types.

Field/Value	Description
RequestIdentifier 0x00000000	Indicates that the package is a Network Information Discovery request from the protocol client to the protocol server.
ResponseIdentifier 0xFFFFFFFF	Indicates that the package is a Network Information Discovery response from the protocol server to the protocol client.

2.2.2 Structures

2.2.2.1 Network Information Discovery Request

The package from protocol clients MUST set the **Id** field to RequestIdentifier and SHOULD include a single byte payload of any value.

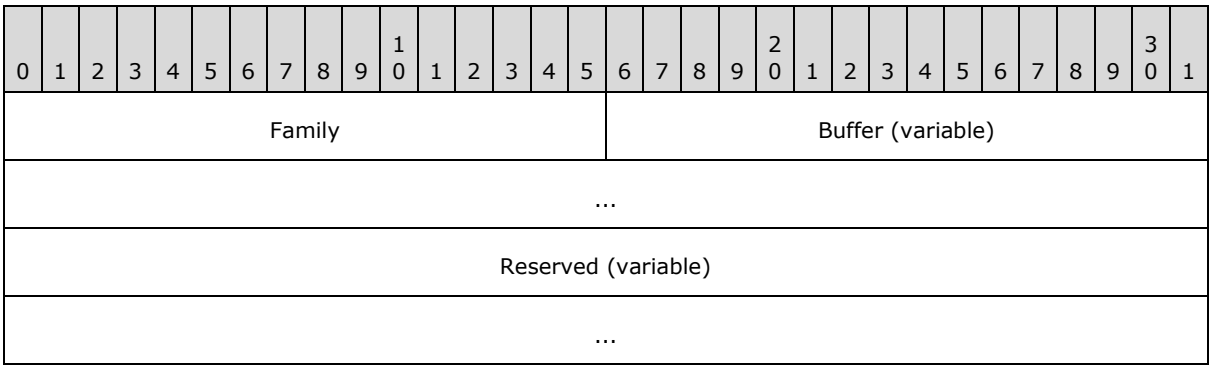


Id (4 bytes): A message identifier to specify the package type of Network Information Discovery Request. The value MUST be RequestIdentifier, which is defined in 2.2.1.1.

Payload (1 byte): A package payload.

2.2.2.2 SOCKADDR_STORAGE

SOCKADDR_STORAGE is a 128-byte structure that is formatted as follows:



Family (2 bytes): The address family of the socket. This field MUST contain one of the following values:

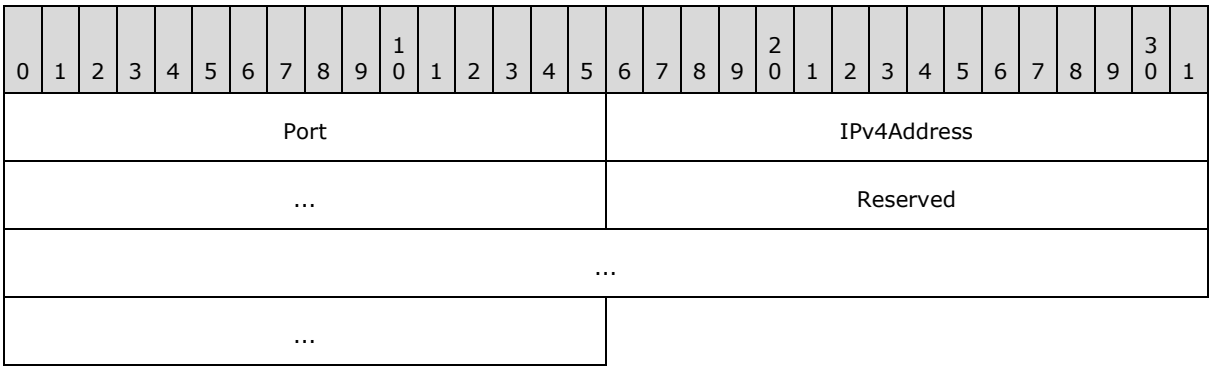
Value	Meaning
InterNetwork 0x0002	When set, this indicates an IPv4 address in the socket.
InterNetworkV6 0x0017	When set, this indicates an IPv6 address in the socket.

Buffer (variable): A variable-length buffer that contains the socket address information. If the value of the **Family** field is 0x0002, this field MUST be interpreted as SOCKADDR_IN (section 2.2.2.2.1). Otherwise, if the value of the **Family** field is 0x0017, this field MUST be interpreted as SOCKADDR_IN6 (section 2.2.2.2.2).

Reserved (variable): The remaining bytes within the size of the SOCKADDR_STORAGE structure (128 bytes) MUST NOT be used and MUST be reserved. The server SHOULD set this to zero, and the client MUST ignore it on receipt.

2.2.2.2.1 SOCKADDR_IN

SOCKADDR_IN is a 14-byte structure formatted as follows. All fields in this structure are in network byte order.



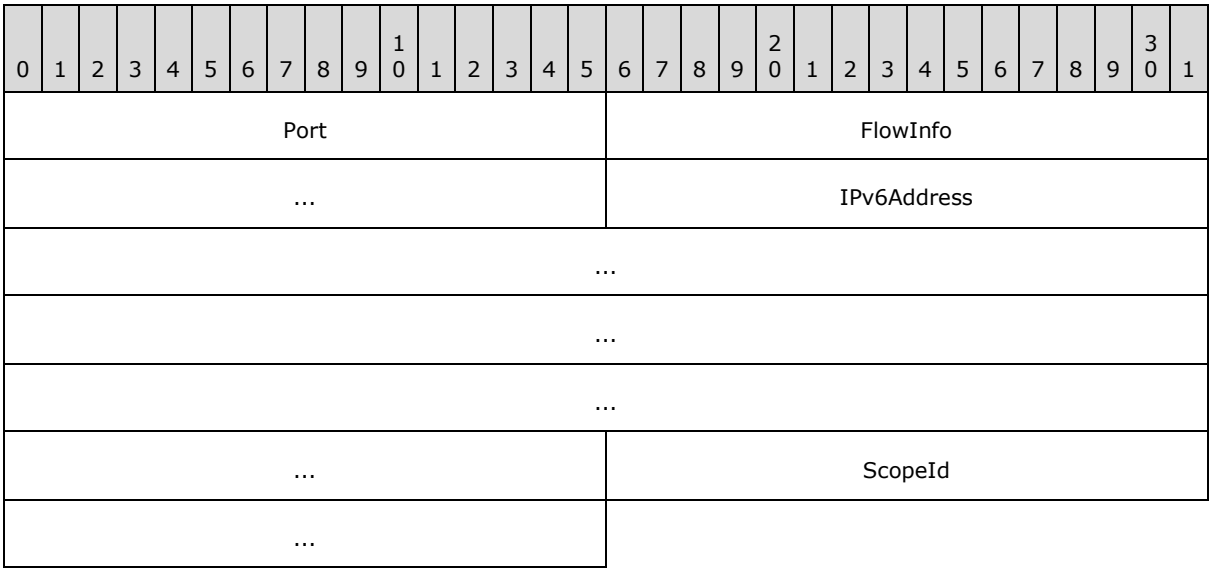
Port (2 bytes): This field MUST NOT be used and MUST be reserved. The server SHOULD set this field to zero, and the client MUST ignore it on receipt.

IPv4Address (4 bytes): The IPv4 address.

Reserved (8 bytes): This field MUST NOT be used and MUST be reserved. The server SHOULD set this field to zero, and the client MUST ignore it on receipt.

2.2.2.2.2 SOCKADDR_IN6

SOCKADDR_IN6 is a 26-byte structure formatted as follows. All fields in this structure are in network byte order.



Port (2 bytes): This field MUST NOT be used and MUST be reserved. The server SHOULD set this field to zero, and the client MUST ignore it on receipt.

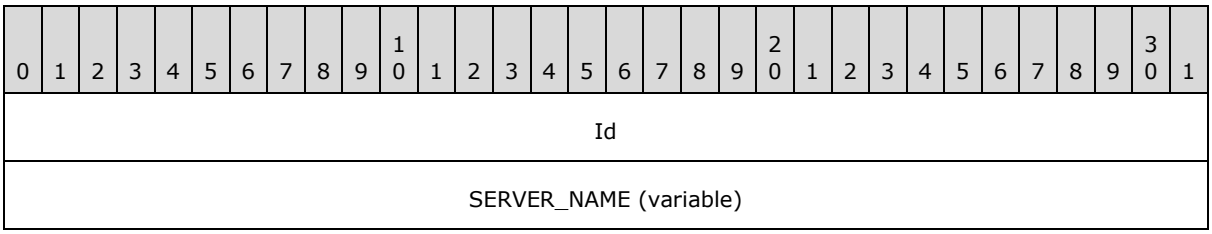
FlowInfo (4 bytes): The server SHOULD set this field to zero, and the client MUST ignore it on receipt.

IPv6Address (16 bytes): IPv6 address.ScopeId (4 bytes): The server SHOULD set this field to zero, and the client MUST ignore it on receipt.

ScopeId (4 bytes): The server SHOULD set this field to zero, and the client MUST ignore it on receipt.

2.2.2.3 Network Information Discovery Response

The package from protocol servers MUST set all fields with the format described in this section. There is no alignment requirement for fields after the variable SERVER_NAME, but all fields after that variable are presented in a specific structure with fixed size.



...
VERSION
LOWEST_VERSION
IPv4_DNS_NUM
IPv4_DNS_ADDRESS (variable)
...
IPv6_DNS_NUM
IPv6_DNS_ADDRESS (variable)
...

Id (4 bytes): Message identifier to specify the package type of Network Information Discovery Response. The value MUST be ResponseIdentifier, which is defined in section 2.2.1.1.

SERVER_NAME (variable): The protocol server's NetBIOS name in a null-terminated Unicode string.

VERSION (4 bytes): The current version of the protocol package. The protocol server MUST set this field according to the corresponding protocol version it is using. The value for this field MUST be 256 or 512. If the field is set to 256, all fields starting from **IPv4_DNS_NUM** are ignored by the protocol client.

LOWEST_VERSION (4 bytes): The lowest version of the protocol package that the protocol server supports. The value of this field MUST be 256 or 512.

IPv4_DNS_NUM (4 bytes): The number of elements stored in **IPv4_DNS_ADDRESS**. This field is set to a value greater than or equal to 0x00000000 if the following IP address fields are used. It is set to 0xFFFFFFFF to ignore all following fields in the message.

IPv4_DNS_ADDRESS (variable): A list of IP addresses stored in the SOCKADDR_STORAGE structure, 128 bytes each. The total number MUST be equal to **IPv4_DNS_NUM**.

IPv6_DNS_NUM (4 bytes): The number of elements stored in **IPv6_DNS_ADDRESS**. The value MUST be greater than or equal to 0x00000000.

IPv6_DNS_ADDRESS (variable): A list of IP addresses stored in the SOCKADDR_STORAGE structure, 128 bytes each. The total number MUST be equal to **IPv6_DNS_NUM**.

2.2.3 Namespaces

None.

2.2.4 Messages

Message	Description
Network information discovery request	The User Datagram Protocol (UDP) broadcast packages are sent from protocol clients to discover protocol servers within the broadcast scope.
Network information discovery response	The UDP packages are sent from protocol servers to protocol clients to respond to the discovery package and to carry the protocol server network information.

2.3 Directory Service Schema Elements

None.

3 Protocol Details

3.1 Client Details

3.1.1 Abstract Data Model

None.

3.1.2 Timers

None.

3.1.3 Initialization

None.

3.1.4 Higher-Layer Triggered Events

None.

3.1.5 Message Processing Events and Sequencing Rules

The protocol client creates a socket on a randomly selected port and sends a Network Information Discovery Request with appropriate Id (section 2.2.1.1) defined and a payload to the IPv4 broadcast address and IPv6 link-local all nodes multicast address.

The protocol client then listens to the same port waiting for the Network Information Discovery Response.

3.1.6 Timer Events

None.

3.1.7 Other Local Events

None.

3.2 Server Details

3.2.1 Abstract Data Model

None.

3.2.2 Timers

None.

3.2.3 Initialization

The protocol server creates a UDP socket that listens to port 8912 when the protocol is started.

3.2.4 Higher-Layer Triggered Events

None.

3.2.5 Message Processing Events and Sequencing Rules

When the server receives a UDP Network Information Discovery request package from port 8912, it checks the **Id** in the package. If the **Id** is not a correct request Id as defined in section 2.2.1.1, the package is ignored. If the **Id** is the correct request **Id** as defined in section 2.2.1.1, the server finds the NetBIOS name and all DNS addresses on its network adapters and then sets the corresponding fields of the Network Information Discovery response package.

3.2.6 Timer Events

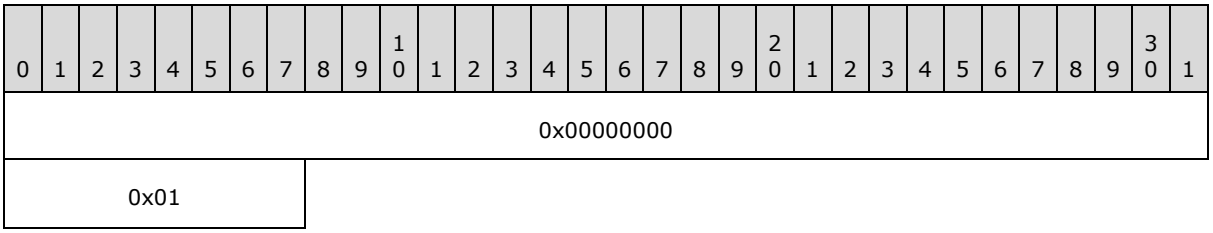
None.

3.2.7 Other Local Events

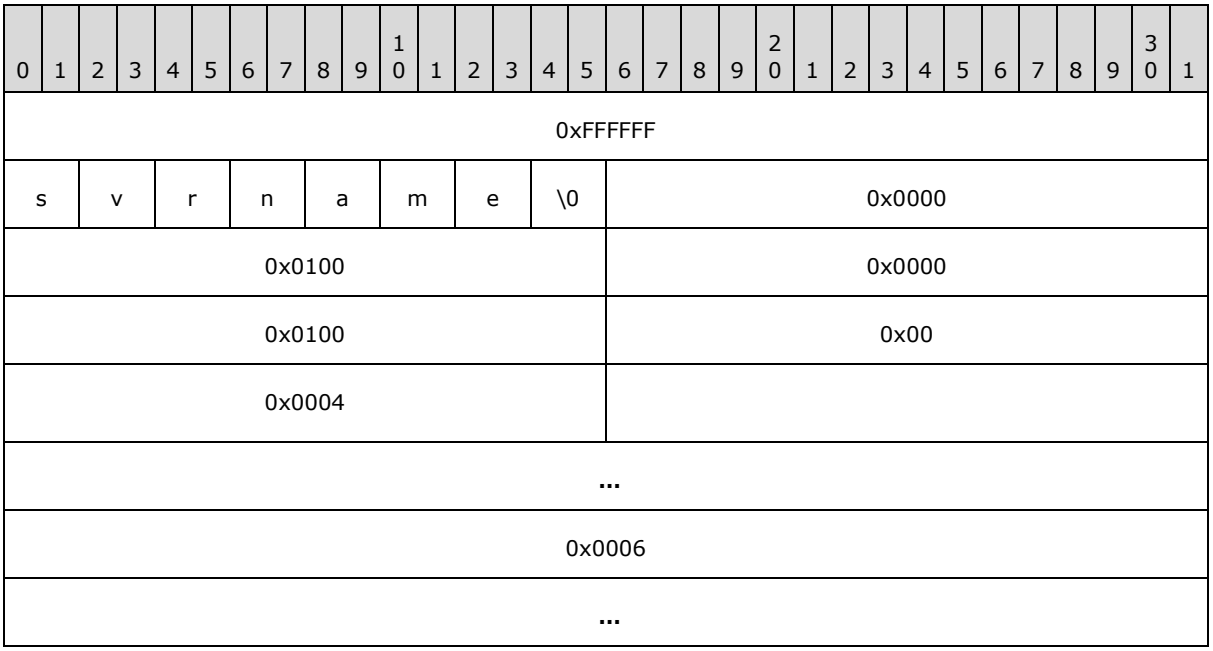
None.

4 Protocol Examples

Network Information Discovery Request



Network Information Discovery Response



The example response package's **IPv4_DNS_ADDRESS** field contains four elements of structure SOCKADDR_STORAGE, and the **IPv6_DNS_ADDRESS** field contains six elements of structure SOCKADDR_STORAGE.

5 Security

5.1 Security Considerations for Implementers

None.

5.2 Index of Security Parameters

None.

6 Appendix A: 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:

- Windows Home Server 2011 server software
- Windows Small Business Server 2011 Essentials
- Windows Server 2012 operating system
- Windows Server 2012 R2 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.

7 Change Tracking

This section identifies changes that were made to the [MS-SNID] protocol document between the August 2013 and November 2013 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
1.5 Prerequisites/Preconditions	69207 Removed MUST from sentence.	Y	Content updated.
1.6 Applicability Statement	69207 Removed MUST from sentence.	Y	Content updated.
2.2.2.1 Network Information Discovery Request	69209 Added description for Payload.	Y	Content updated.
2.2.2.2 SOCKADDR_STORAGE	69214 Added section.	Y	New content added.
2.2.2.2.1 SOCKADDR_IN	69214 Added section.	Y	New content added.
2.2.2.2.2 SOCKADDR_IN6	69214 Added section.	Y	New content added.
2.2.2.3 Network Information Discovery Response	69210 Changed the term beacon to protocol package.	Y	Content updated.
3.1 Client Details	69208 Revised title.	Y	Content updated.
3.2 Server Details	69208 Revised title.	Y	Content updated.

Section	Tracking number (if applicable) and description	Major change (Y or N)	Change type
3.2.5 Message Processing Events and Sequencing Rules	69215 ADDED behavior if the Id is NOT A correct request/response Id.	Y	Content updated.
6 Appendix A: Product Behavior	69216 Updated product applicability list.	N	Content updated.

8 Index