[MS-NBTE]:
NetBIOS over TCP (NetBT) Extensions

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.

Copyright © 2013 Microsoft Corporation.

Revision Summary

Date	Revision History	Revision Class	Comments
05/22/2009	0.1	Major	First Release.
07/02/2009	0.1.1	Editorial	Revised and edited the technical content.
08/14/2009	1.0	Major	Updated and revised the technical content.
09/25/2009	1.0.1	Editorial	Revised and edited the technical content.
11/06/2009	1.0.2	Editorial	Revised and edited the technical content.
12/18/2009	1.0.3	Editorial	Revised and edited the technical content.
01/29/2010	2.0	Major	Updated and revised the technical content.
03/12/2010	3.0	Major	Updated and revised the technical content.
04/23/2010	4.0	Major	Updated and revised the technical content.
06/04/2010	4.0.1	Editorial	Revised and edited the technical content.
07/16/2010	5.0	Major	Significantly changed the technical content.
08/27/2010	5.0	No change	No changes to the meaning, language, or formatting of the technical content.
10/08/2010	6.0	Major	Significantly changed the technical content.
11/19/2010	6.0	No change	No changes to the meaning, language, or formatting of the technical content.
01/07/2011	7.0	Major	Significantly changed the technical content.
02/11/2011	7.0	No change	No changes to the meaning, language, or formatting of the technical content.
03/25/2011	8.0	Major	Significantly changed the technical content.
05/06/2011	9.0	Major	Significantly changed the technical content.
06/17/2011	9.1	Minor	Clarified the meaning of the technical content.
09/23/2011	9.1	No change	No changes to the meaning, language, or formatting of the technical content.
12/16/2011	10.0	Major	Significantly changed the technical content.
03/30/2012	10.0	No change	No changes to the meaning, language, or formatting of the technical content.
07/12/2012	11.0	Major	Significantly changed the technical content.
10/25/2012	12.0	Major	Significantly changed the technical content.

Copyright © 2013 Microsoft Corporation.

Date	Revision History	Revision Class	Comments
01/31/2013	12.0	No change	No changes to the meaning, language, or formatting of the technical content.
08/08/2013	13.0	Major	Significantly changed the technical content.

Copyright © 2013 Microsoft Corporation.

Contents

1 Introduction 6
1.1 Glossary 6
1.2 References 6
1.2.1 Normative References 7
1.2.2 Informative References 7
1.3 Overview 7
1.4 Relationship to Other Protocols 8
1.5 Prerequisites/Preconditions 8
1.6 Applicability Statement 8
1.7 Versioning and Capability Negotiation 8
1.8 Vendor-Extensible Fields 8
1.9 Standards Assignments 8

2 Messages 9
2.1 Transport 9
2.2 Message Syntax 9
2.2.1 NetBIOS Name Syntax 9
2.2.2 MULTIHOMED NAME REGISTRATION REQUEST 9
2.2.3 LMHOSTS File Syntax 9
2.2.3.1 Predefined Keywords in LMHOSTS File 10
2.2.3.2 Creating Lmhosts Entries for Specific NetBIOS Names 10

3 Protocol Details 12
3.1 NetBIOS End Node Details 12
3.1.1 Abstract Data Model 12
3.1.2 Timers 12
3.1.3 Initialization 13
3.1.4 Higher-Layer Triggered Events 13
3.1.4.1 Registering a NetBIOS Name 13
3.1.4.2 Resolving a NetBIOS Name 14
3.1.4.2.1 NetBIOS Name Server Selection 14
3.1.5 Message Processing Events and Sequencing Rules 14
3.1.5.1 Handling a NAME REGISTRATION REQUEST 14
3.1.6 Timer Events 15
3.1.7 Other Local Events 15
3.1.8 Using the LMHOSTS File to Resolve a Name Query 15
3.1.8.1 Using a #include LMHOSTS File 16
3.1.8.2 Alternate Block Processing 16
3.2 NetBIOS Name Server Details 16
3.2.1 Abstract Data Model 16
3.2.2 Timers 16
3.2.3 Initialization 16
3.2.4 Higher-Layer Triggered Events 16
3.2.5 Message Processing Events and Sequencing Rules 16
3.2.5.1 Handling a NAME REGISTRATION REQUEST (GROUP) 16
3.2.5.2 Handling a MULTIHOMED NAME REGISTRATION REQUEST (GROUP) 17
3.2.5.3 Handling a MULTIHOMED NAME REGISTRATION REQUEST (UNIQUE) 17
3.2.6 Timer Events 17
3.2.7 Other Local Events 17

Copyright © 2013 Microsoft Corporation.

4 Protocol Examples 18
4.1 NetBIOS Registration Example 18
4.2 LMHOSTS File Example 20

5 Security Considerations 21
5.1 Security Considerations for Implementers 21
5.2 Index of Security Parameters 21

6 Appendix A: Product Behavior 22

7 Change Tracking 24

8 Index 26

Copyright © 2013 Microsoft Corporation.

1 Introduction

The NetBIOS over TCP (NetBT) Extensions specify the extensions to the NetBIOS over TCP (NetBT) protocol, as specified in [RFC1001] and [RFC1002]. These extensions modify the syntax of allowable NetBIOS names and the behavior of timers, and add support for multihomed hosts.

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]:

code page
Internet host name
NetBIOS name
NetBIOS Name Server (NBNS)
NetBIOS over TCP/IP (NetBT)
Windows Internet Name Service (WINS)
Universal Naming Convention (UNC)

The following terms are specific to this document:

group name: As defined in [RFC1002], a NetBIOS name that can be owned by any number of nodes.

LMHOST: A text file that contains entries that individually map a computer name or a NetBIOS service name to an IPv4 address. The LMHOST file is consulted when normal NetBIOS name resolution protocols fail on the wire. This legacy file is no longer installed by default on Windows systems. LM stands for "LAN Manager".

multihomed: Having two or more network interfaces on which NetBIOS over TCP is enabled.

unique name: As defined in [RFC1002], a NetBIOS name that can be owned by a single node.

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 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].

Copyright © 2013 Microsoft Corporation.

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.

[HYBRID] Noon, F., "HYBRID NETBIOS END-NODES", April 1993, http://tools.ietf.org/id/draft-noon-hybrid-netbios-01.txt

[RFC1001] Network Working Group, "Protocol Standard for a NetBIOS Service on a TCP/UDP Transport: Concepts and Methods", STD 19, RFC 1001, March 1987, http://www.ietf.org/rfc/rfc1001.txt

[RFC1002] Network Working Group, "Protocol Standard for a NetBIOS Service on a TCP/UDP Transport: Detailed Specifications", STD 19, RFC 1002, March 1987, http://www.ietf.org/rfc/rfc1002.txt

[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

[RFC2132] Alexander, S., and Droms, R., "DHCP Options and BOOTP Vendor Extensions", RFC 2132, March 1997, http://www.ietf.org/rfc/rfc2132.txt

1.2.2 Informative References

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

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

[RFC1035] Mockapetris, P., "Domain Names - Implementation and Specification", STD 13, RFC 1035, November 1987, http://www.ietf.org/rfc/rfc1035.txt

[RFC4795] Aboba, B., Thaler, D., and Esibov, L., "Link-Local Multicast Name Resolution (LLMNR)", RFC 4795, January 2007, http://www.ietf.org/rfc/rfc4795.txt

1.3 Overview

NetBIOS resources are referenced by **NetBIOS name**. An application, representing a resource, registers one or more NetBIOS names that it uses to communicate with other hosts on the network. This document does the following:

1. It discusses NetBIOS name registration and name querying on hosts with more than one interface.

2. When a name server receives a registration for a **group name**, [RFC1002] requires that the name server replace any existing entry with the new entry, so that only one IP address can be registered for a group name. However, a group name is one that can be owned by any number of nodes. This document modifies the behavior of group name registrations to allow the name server to keep multiple addresses.

3. Defines the format of the LMHOSTS file and when it is accessed. The LMHOSTS file is read on two separate occasions. At the initialization of the NetBIOS system, LMHOSTS is read to initialize the local name cache with the entries that are labeled "#PRE". During NetBIOS name resolution, if the name cannot be resolved from the local name cache or by using the normal NetBIOS protocol

Copyright © 2013 Microsoft Corporation.

name resolutions, then the NetBIOS name resolution process system can be configured to scan the LMHOSTS file on a per-query basis, looking for entries that resolve the query.

1.4 Relationship to Other Protocols

A NetBIOS name may or may not be derived from an **Internet host name**. The syntax for an Internet host name is much more constrained than the syntax for an arbitrary NetBIOS name. Therefore, it is possible to derive a NetBIOS name from a given Internet host name, but not necessarily vice versa.

NetBIOS is used to resolve names within a local subnet, and is also used to resolve names within a larger network using a name server. However, it is only defined for IPv4. As such, its use for name resolution has largely been superseded by newer protocols, such as the Link-Local Multicast Name Resolution (LLMNR) Protocol [RFC4795] and the Domain Name System (DNS) [RFC1035].

1.5 Prerequisites/Preconditions

The prerequisites and preconditions are unchanged from [RFC1001] and [RFC1002].

1.6 Applicability Statement

This extension is applicable for discovering the IPv4 addresses of resources.

1.7 Versioning and Capability Negotiation

There is no versioning or localization support in this extension.

1.8 Vendor-Extensible Fields

It is important to understand that the choice of name used by a higher-layer protocol or application is up to that protocol or application and not NetBIOS. As such, this section provides a convention for use by higher-layer protocols and applications, but the extensions in this document do not enforce the use of this convention.

The recommended convention is for higher-layer protocols and applications to use the first 15 bytes of the Internet host name of the machine (padded with spaces if shorter than 15 bytes) followed by a 1-byte NetBIOS suffix chosen by the higher-layer protocol or application.<1>

The recommended convention allows for 256 NetBIOS suffix values and vendors can define a new value. However, there is no mechanism to acquire a unique value and hence collisions are possible if multiple vendors define the same NetBIOS suffix values. It is up to each higher-layer protocol or application to specify what NetBIOS suffix it uses, or how the NetBIOS name is constructed if it does not use this recommended convention.

1.9 Standards Assignments

None beyond what is in [RFC1001], [RFC1002], and [RFC2132] section 8.7.

Copyright © 2013 Microsoft Corporation.

2 Messages

2.1 Transport

The transport is unchanged from [RFC1002] except that name resolution is supported only over UDP and not TCP. The term "NetBIOS over TCP" refers to the standard protocol in the same way as [RFC1001] and [RFC1002] do; that is, "TCP" refers to "TCP/IP".

2.2 Message Syntax

2.2.1 NetBIOS Name Syntax

[RFC1001] and [RFC1002] are confusing with respect to the definition of the name syntax. [RFC1001] section 5.2 states: "The name space is flat and uses sixteen alphanumeric characters. Names may not start with an asterisk (*)."

[RFC1002] section 4.1 states: "The following is the uncompressed representation of the NetBIOS name "FRED", which is the 4 ASCII characters, F, R, E, D, followed by 12 space characters (0x20)."

It should be clear from the previous statement, because an asterisk and space characters are not letters or numbers, that the term "alphanumeric characters" is confusing at best.

This document clarifies the ambiguity by specifying that the name space is defined as sixteen 8-bit binary bytes, with no restrictions, except that the name SHOULD NOT<2><3> start with an asterisk (*).

Neither [RFC1001] nor [RFC1002] discusses whether names are case-sensitive. This document clarifies this ambiguity by specifying that because the name space is defined as sixteen 8-bit binary bytes, a comparison MUST be done for equality against the entire 16 bytes. As a result, NetBIOS names are inherently case-sensitive.

It is important to understand that the choice of name used by a higher-layer protocol or application is up to that protocol or application and not NetBIOS. A NetBIOS over TCP implementation MUST NOT enforce the use of the convention discussed in section 1.8.

2.2.2 MULTIHOMED NAME REGISTRATION REQUEST

[RFC1002] section 4.2.2 defines the format of a NAME REGISTRATION REQUEST. This extension adds a MULTIHOMED NAME REGISTRATION REQUEST with an identical format except that the OPCODE field MUST be set to 0xF (15).

2.2.3 LMHOSTS File Syntax

The **LMHOSTS** file is a static text file of NetBIOS name and IPv4 addresses along with additional directives for processing, including a "#include <filename>" mechanism.

- There can be one entry per line. An entry consists of an IPv4 address and a name, which can be either a computer name or a NetBIOS service name.
- Comment lines in the LMHOSTS file start with "#".
- Comments can start after the start of a line, with "#", and without the use of LMHOSTS keywords. (See the LMHOSTS keywords in the table that follows.)
- The "#" can also denote LMHOSTS keywords listed in the table that follows.

Copyright © 2013 Microsoft Corporation.

- ComputerName Entries consist of an IPv4 address and a NetBIOS computer name where the name is 1 to 15 characters in length. A computer name can be used to either: 1) resolve a name to an IP address, or 2) resolve a NetBIOS service name to an address. Example: "131.107.7.29 emailsrv1".
- ServiceName entries consist of an IPv4 address and a NetBIOS service name that specifies a 16-byte name where the last byte indicates the type of the service and bytes 1 to 15 specify ComputerName, padded at the end with blanks to the 15th byte:
  - 131.107.7.30 "ComputerName \0x03" where the last byte is specified in hex.
- Entry Names are not case-sensitive.
- The LMHOSTS file has an implementation-specific file location.<4>

2.2.3.1 Predefined Keywords in LMHOSTS File

The LMHOSTS file can contain predefined keywords that are prefixed with the "#" character. The following LMHOSTS keywords table lists possible LMHOSTS keywords.

LMHOSTS Keyword	Description
#PRE	A tag that can follow the name in an entry. Tagged entries are loaded as permanent entries in the NetBIOS name cache during initialization of the NetBIOS name system. Preloaded entries are used to reduce network broadcasts. An entry tagged with #PRE will be loaded in the Local Name Table.
#DOM:DomainName	A tag that can follow the name in an entry. It identifies Domain Controllers for the given domain name.
#INCLUDE Path\FileName	Reads entries in the file "Path\FileName". FileName may be a local filesystem path or a **Universal Naming Convention (UNC)** path as specified in [MS-DTYP] section 2.2.57. There must be entries for the computer names of remote servers hosting the shares in the local LMHOSTS file; otherwise, the shares will not be accessible.
#BEGIN_ALTERNATE and #END_ALTERNATE	A tag defines a list of alternative locations for the LMHOSTS files. This is used as a reliability mechanism. Only one of the files in a "#BEGIN"/"#END" block will be used. An attempt is made to read a file, one file at a time.
#MH	A tag that can follow the name of an entry. If present, that name can have multiple IP addresses reflected in multiple entries with the same IP address. This allows the reading of an LMHOSTS file to continue after a successful match of an entry.

2.2.3.2 Creating Lmhosts Entries for Specific NetBIOS Names

A ComputerName in the LMHOSTS file can be up to 15 bytes in length. Names shorter than 15 bytes are padded with spaces.

However, it might be necessary to resolve a specific 16-byte NetBIOS name to a NetBIOS application running on a remote computer. Any arbitrary 16-byte NetBIOS name can be configured in the LMHOSTS file by using the following syntax:

IPv4Address "<Name><SpacePadding>\0xNN"

Copyright © 2013 Microsoft Corporation.

In which:

- IPv4Address is the IPv4 address to which this NetBIOS name is resolved.
- <Name> is the first part of the NetBIOS name (up to 15 bytes).
- <SpacePadding> is needed to ensure that the full NetBIOS name is 16 bytes. If the Name portion has fewer than 15 bytes, it MUST be padded with spaces up to 15 bytes.
- \0xNN indicates the two-digit hexadecimal representation of the 16th byte of the NetBIOS name. The syntax \0xNN can represent any byte in the NetBIOS name but is most often used for the 16th character.

Copyright © 2013 Microsoft Corporation.

3 Protocol Details

3.1 NetBIOS End Node 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.

The data model is as specified in [RFC1002] and [HYBRID], with the following clarifications.

- **Interface List:** A list of interfaces on which NetBIOS over TCP is enabled, in order from most preferred to least preferred. This list SHOULD be administratively configurable. Each entry contains the following:
  - **NetBIOS Name Service:** A configured list of NetBIOS name server addresses, in order from most preferred to least preferred.
- **Node Type:** The NetBIOS node type as specified in [RFC1002] and [HYBRID]. This document clarifies that this state is global, not per-interface.
- **Query Table:** For each outstanding resolution in progress, in addition to the information specified in [RFC1002] and [HYBRID], the following field is kept:
  - **Interface List:** A list of interfaces awaiting a response.
- **Local Name Table:** The local name table as specified in [RFC1002] and [HYBRID], with the following clarifications:
  - **Interface List:** A list of interfaces on which registration was attempted. Each entry also contains:
    - **Conflict Detected Flag:** A flag that, if TRUE, indicates that a conflict was seen. This document clarifies that this flag is kept per-interface (not globally) for each local name.
  - **Refresh Timeout:** [RFC1002] section 5.1.2.1 specifies the use of a Refresh Timeout. This document clarifies that this state is global, not per-interface.
- **ReadLMHostsFile:** A Boolean value that SHOULD default to FALSE.<5> If TRUE, the LMHOSTS file will be read if the LMHOSTS file exists.
- **LMHostsFileLocation:** The location of the LMHOSTS file.

3.1.2 Timers

There is an lmhost_include timer used to determine whether an LMHOSTS file can be read (see section 3.1.6).

There are timers specified in [RFC1002] and [HYBRID], except as follows.

[RFC1002] section 6 states that UCAST_REQ_RETRY_TIMEOUT "should" be 5 seconds but that an adaptive timer "may" be used. In these extensions, UCAST_REQ_RETRY_TIMEOUT SHOULD be set to 1.5 seconds.

Copyright © 2013 Microsoft Corporation.

[RFC1002] specifies the use of a periodic REFRESH_TIMER for each entry in the Local Name Table, with a period of Refresh Timeout. This extension clarifies that the REFRESH_TIMER for each name is kept globally, not per-interface.

3.1.3 Initialization

The rules for initialization are specified in [RFC1002] and [HYBRID]. However, they are ambiguous as to how an end node chooses a node type. This document clarifies the rules as follows.

The NetBIOS Node Type (see [RFC1001] section 10 and [HYBRID]) SHOULD be administratively configurable, and be set to H by default.

If DHCP is in use and a NetBIOS over TCP/IP Node Type Option (see [RFC2132] section 8.7) is provided by the DHCP server, an end node MUST set its Node Type to the value indicated in the option. If this DHCP option is obtained over multiple interfaces, then the end node MUST choose one of them in any implementation-specific<6> way.

ReadLMHostsFile SHOULD be set from an implementation-specific store for its value.<7>

If **ReadLMHostsFile** is TRUE and if the LMHOSTS file exists, the LMHOSTS file MUST be read at NetBIOS initialization, and any entries marked with #PRE MUST be loaded into the Local Name Table (see section 3.1.8). Before attempting to open an LMHOSTS file from the location specified by **LMHostsFileLocation**, the lmhost_include timer MUST be initialized to 6 seconds.

The NetBT system MAY read the centralized LMHOSTS file before a user logs on to the computer as part of NetBT initialization.<8><9>

3.1.4 Higher-Layer Triggered Events

Except as specified in the following sections, the rules for handling higher-layer triggered events are as specified in [RFC1002] section 5.1, and [HYBRID]. This document clarifies that whenever [RFC1002] or [HYBRID] specify that a packet is to be broadcast, the end node MUST broadcast the packet on all interfaces in its Interface List unless otherwise specified.

3.1.4.1 Registering a NetBIOS Name

When a higher-layer protocol or application requests that a NetBIOS name be registered on a given interface, processing MUST be done as specified in [RFC1002] section 5.1 and [HYBRID] according to its Node Type, except as follows.

If the name already exists in the Local Name Table and the Conflict Detected flag is set on any interface in the Interface List, then the node MUST immediately fail the request.

If the end node is **multihomed**, the name to be registered is unique, and the end node is configured with a NetBIOS name server, then the end node SHOULD send a MULTIHOMED NAME REGISTRATION REQUEST (UNIQUE).

If the registration completes successfully and no entry exists in the Local Name Table, then one MUST be added with the Interface List set to contain the given interface, with its Conflict Detected flag cleared. The Refresh Timeout MUST be set to the TTL in the POSITIVE NAME REGISTRATION RESPONSE, or to 5 minutes if the TTL is less than 5 minutes.

If the registration completes successfully and an entry already exists in the Local Name Table, then the given interface MUST be added to the entry's Interface List, with its Conflict Detected flag cleared. The Refresh Timeout MUST then be set, unless its value would increase by doing so, to the

Copyright © 2013 Microsoft Corporation.

TTL in the POSITIVE NAME REGISTRATION RESPONSE, or to 5 minutes if the TTL is less than 5 minutes.

If the registration fails and an entry already exists in the Local Name Table, then the given interface MUST be added to the entry's Interface List, with its Conflict Detected flag set.

If the name begins with an asterisk (*), then the request MUST be completed successfully without attempting to register the name or check for conflicts.

3.1.4.2 Resolving a NetBIOS Name

The rules for resolving a NetBIOS name are unchanged from [RFC1002] section 5.1 and [HYBRID] except as follows.

For Node Types other than B, the list of NetBIOS name servers to use MUST first be constructed as specified in section 3.1.4.2.1.

Name queries MUST then be performed as specified in [RFC1002] section 5.1 and [HYBRID], except that instead of simply querying a single NetBIOS name server, each name server in the list of NetBIOS name servers **(NBNS)** MUST be consulted in turn until one responds or the end of the list is reached.

If the end of the list of NBNS is reached, the Name Query will be processed against a local file LMHOSTS (see section 3.1.8) if **ReadLMHostsFile** is TRUE.

3.1.4.2.1 NetBIOS Name Server Selection

If the application or higher-layer protocol specified a specific interface, then the NetBIOS name server list MUST be the list of name servers for that interface.

If the application or higher-layer protocol did not specify a specific interface, then the NetBIOS name server list MUST be formed by concatenating the lists of name servers for each interface in the Interface List, in the order the interfaces appear in the Interface List.

3.1.5 Message Processing Events and Sequencing Rules

The rules for processing NetBIOS messages are unchanged from [RFC1002] section 5.1 and [HYBRID]. This document clarifies that whenever [RFC1002] or [HYBRID] specify that a packet is to be broadcast, the end node MUST broadcast the packet on all interfaces in its Interface List, unless otherwise specified. In addition, whenever [RFC1002] or [HYBRID] state that the Conflict Detected flag is set, this refers to the Conflict Detected flag for the interface over which the relevant message was received, unless otherwise specified.

3.1.5.1 Handling a NAME REGISTRATION REQUEST

[RFC1002] section 5.1 and [HYBRID] are somewhat confusing as to how a node is to respond to a NAME REGISTRATION REQUEST when the name matches an entry in the local name table with the Conflict Detected flag set. For example, [RFC1002] section 5.1.1.5 states that a NEGATIVE NAME REGISTRATION RESPONSE is sent if an entry exists in the local name table. However, it later clarifies that a name in the state "conflict detected" does not "logically" exist on that node and that such an entry will not be used for purposes of processing incoming request packets.

This document clarifies that a node MUST NOT send a NEGATIVE NAME REGISTRATION RESPONSE if there exists any entry in the name's Interface List whose Conflict Detected flag is set, independent of the interface on which the NAME REGISTRATION REQUEST was received.

Copyright © 2013 Microsoft Corporation.

In addition, the node MUST NOT send a NEGATIVE NAME REGISTRATION RESPONSE if the name begins with an asterisk ("*").

3.1.6 Timer Events

If the lmhost_include timer expires, the file MUST be closed and the query completed with an error. See Alternate Block Processing (section 3.1.8.2).

3.1.7 Other Local Events

When an address change occurs on an interface, then the node MUST do the following.

For each entry in the Local Name Table, if the interface is in the entry's Interface List, then the node MUST repeat the registration of that name on that interface, and update the interface's Conflict Detected flag to be clear if it completes successfully, or set if it fails.

3.1.8 Using the LMHOSTS File to Resolve a Name Query

When NetBIOS name resolution, which uses NetBIOS protocols, does not result in successful name resolution, and if **ReadLMHostsFile** is TRUE and the LMHOSTS file exists, the ComputersQuery MUST read the LMHOSTS file for NetBIOS name resolution to an IPv4 address.

When resolving a name, the LMHOSTS file MUST be opened and read, and a search is made for a matching entry, or entries, so that name resolution can return a list of IP addresses. The LMHOSTS file MUST be read sequentially, matching the name in the query with the name of an entry read from the LMHOSTS file, until all matches are found or no additional entries are in the LMHOSTS file. The matching function works as follows:

1. Create an empty list of matching IP addresses.
2. If the 16th byte of the query name is 0x1c, then search the list of domain names (LMHOSTS entries with the #DOM qualifier) that were preloaded in the Local Name Table during NetBIOS initialization. If the query name matches a domain entry, then add that entry's IP address to the list and further processing MUST stop.
3. Search the Local Name Table for a match of names preloaded during NetBIOS initialization. If the query name matches an entry, then add that entry's IP address to the list and further processing MUST stop.
4. Read a line from the LMHOSTS file.
5. If the NetBIOS name from the LMHOSTS file is less than 16 bytes in length, pad the name with spaces, and uppercasing all characters within the ASCII range which results in ComputerName. If the resulting ComputerName and the query name match, then add the entry's IP address to the list of matching IP addresses. If there is no #MH tag on the entry, then further processing of the LMHOSTS file MUST stop.
6. If the NetBIOS name from the LMHOSTS file is exactly 16 bytes in length then this results in ComputerName. If the resulting ComputerName and the query name, then add the entry's IP address to the list of matching IP addresses. If there is no #MH tag on the entry, then further processing of the LMHOSTS file MUST stop.
7. If there are more entries from the LMHOSTS file to be read, loop to step four.

Name resolution returns the list of matching IP addresses.

Copyright © 2013 Microsoft Corporation.

The #include facility in LMHOSTS can result in reading additional files.

3.1.8.1 Using a #include LMHOSTS File

NetBT can read LMHOSTS files that are located on other computers. This allows the use of a centralized LMHOSTS file that can be accessed through a computer's local LMHOSTS file. Using a centralized LMHOSTS file still requires each computer to have a local LMHOSTS file.

To access a centralized LMHOSTS file, a computer's local LMHOSTS file can have an entry with the #INCLUDE tag for the location of the centralized file. Example:

```
#INCLUDE \\Fileserver\Public\Lmhosts
```

In this example, NetBT includes the LMHOSTS file on the public shared folder of the server. An implementation MUST detect circular #INCLUDE directives and stop any further processing of the file and of the query.

3.1.8.2 Alternate Block Processing

The #BEGIN_ALTERNATE and #END_ALTERNATE tags allow a block of remote LMHOSTS file locations in the reading of the LMHOSTS file. This technique is known as block inclusion.

The files inside an ALTERNATE block MUST be opened and read one at a time and in order.

3.2 NetBIOS Name Server Details

3.2.1 Abstract Data Model

The data model is unchanged from [RFC1002] and [HYBRID]. This document clarifies that a name server MUST support storing at least 25 addresses per NetBIOS name.

3.2.2 Timers

None beyond what is specified in [RFC1002] and [HYBRID].

3.2.3 Initialization

The rules for initialization are unchanged from [RFC1002] and [HYBRID].

3.2.4 Higher-Layer Triggered Events

None.

3.2.5 Message Processing Events and Sequencing Rules

Except as specified in the following sections, the rules for processing NetBIOS messages are unchanged from [RFC1002] section 5.1.4.

3.2.5.1 Handling a NAME REGISTRATION REQUEST (GROUP)

When a name server receives a NAME REGISTRATION REQUEST for a group name, the server MUST handle it as specified in [RFC1002] section 5.1.4.1, except as follows.

Copyright © 2013 Microsoft Corporation.

If an entry exists for a group name, then the name server SHOULD<10> skip the step of removing any previously stored address, so that the new address gets appended to the list rather than replacing it; the name server may choose to store the broadcast address 255.255.255.255 instead.

When appending a new address to an existing list, if the list would become larger than the maximum number of entries per name supported by the name server, the name server MUST remove the oldest stored address and then append the new address. (This is consistent with the [RFC1002] behavior, except that an implementation of [RFC1002] without these extensions only supports a maximum of a single address.)

3.2.5.2 Handling a MULTIHOMED NAME REGISTRATION REQUEST (GROUP)

When a name server receives a MULTIHOMED NAME REGISTRATION REQUEST for a group name, the server MUST handle it the same as a NAME REGISTRATION REQUEST for a group name.

3.2.5.3 Handling a MULTIHOMED NAME REGISTRATION REQUEST (UNIQUE)

When a name server receives a MULTIHOMED NAME REGISTRATION REQUEST for a **unique name**, the server MUST handle it the same as a NAME REGISTRATION REQUEST for a unique name as specified in [RFC1002] section 5.1.4.1, except as follows.

Instead of removing an address when one is already stored, the oldest address MUST only be removed when the maximum number of addresses per name would be exceeded.

3.2.6 Timer Events

None beyond what is specified in [RFC1002] and [HYBRID].

3.2.7 Other Local Events

None beyond what is specified in [RFC1002] and [HYBRID].

Copyright © 2013 Microsoft Corporation.

4 Protocol Examples

4.1 NetBIOS Registration Example

Consider a multihomed node A with an Internet host name of "EXAMPLE" supporting the extensions defined herein.

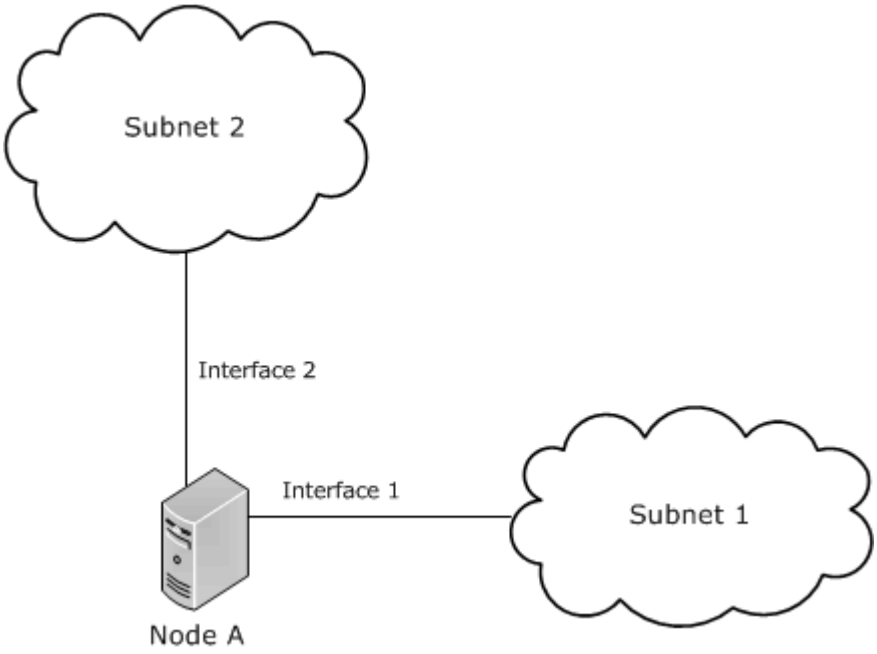

Figure 1: Multihomed node example

1. At startup, Node A uses DHCP on each interface. On interface 1, it gets a NetBIOS over TCP/IP Node Type Option indicating it should be an H node. From interface 2, it gets no DHCP response, but defaults to H node behavior. Hence when it decides to be an H node based on interface 2, it overwrites its Node Type to be the type from interface 2. However, in this example, the value is effectively unchanged because it was already an H node based on DHCP from interface 1.

2. Later, an application wants to publish a NetBIOS name and chooses to use the convention defined in section 1.8. The application chooses a NetBIOS suffix of 0x19, and constructs the NetBIOS name of "EXAMPLE", padded with spaces to 15 bytes, and puts the NetBIOS suffix in the 16th byte, resulting in the following hexadecimal bytes: [0x45, 0x58, 0x41, 0x4D, 0x50, 0x4c, 0x45, 0x20, 0x20, 0x20, 0x20, 0x20, 0x20, 0x20, 0x20, 0x19]. The application asks NetBIOS to register the name EXAMPLE on interface 1, and NetBIOS sends a NAME REGISTRATION REQUEST to the first name server learned on that interface.

   The name server responds with a NEGATIVE NAME REGISTRATION RESPONSE.

   The node receives the response, and EXAMPLE is not added to the Local Name Table, and a failure is returned to the application.

3. The application then tries to register the NetBIOS name EXAMPLE on interface 2.

   No name servers are known on interface 2, so the node falls back to broadcast, and registration succeeds.

Copyright © 2013 Microsoft Corporation.

The name EXAMPLE is then added to the Local Name Table, with interface 2 in the Interface List.

4. Later, another application tries to register the NetBIOS name EXAMPLE on interface 2. The name is already registered on that interface, so a success is immediately returned to the application.

5. Later, another application tries to register EXAMPLE on interface 1.

   The name server again responds with a NEGATIVE NAME REGISTRATION RESPONSE causing registration to again fail on subnet 1.

   This time, because an entry already exists in the Local Name Table, interface 1 is added to the Interface List for the name EXAMPLE in the Local Name Table, with the Conflict Detected flag set.

6. Later, another application tries to register EXAMPLE on interface 1.

   Registration fails immediately (without sending any request) because an entry already exists with the Conflict Detected flag set for some interface (interface 1).

7. Later, another application tries to register EXAMPLE on interface 2.

   Registration fails immediately (without sending any request) because an entry already exists with the Conflict Detected flag set for some interface (interface 1).

8. Later, a NAME QUERY is received on interface 2 for the name EXAMPLE.

   The node replies with a POSITIVE NAME QUERY RESPONSE because the Conflict Detected flag is clear for that interface.

9. Later, a NAME QUERY is received on interface 1 for the name EXAMPLE.

   No response is sent because the Conflict Detected flag is set for that interface.

10. Later, a NAME REGISTRATION REQUEST is received for the name EXAMPLE on interface 1.

    No response is sent because the Conflict Detected flag is set on some interface (interface 1).

11. Later, a NAME REGISTRATION REQUEST is received for the name EXAMPLE on interface 2.

    No response is sent because the Conflict Detected flag is set on some interface (interface 1).

12. Later, an address change occurs on interface 1 after the conflicting entry has been removed from the name server by an administrator.

    Node A tries to reregister the name EXAMPLE on interface 1 by sending a new NAME REGISTRATION REQUEST.

    This time registration succeeds and the server replies with a POSITIVE NAME REGISTRATION RESPONSE.

    The node receives the response and clears the Conflict Detected flag for interface 1.

13. Later, a NAME REGISTRATION REQUEST is received for the name EXAMPLE on interface 1.

    The node replies with a NEGATIVE NAME REGISTRATION RESPONSE because the name is in the Local Name Table and no Conflict Detected flag is set.

14. Later, a NAME REGISTRATION REQUEST is received for the name EXAMPLE on interface 2.

Copyright © 2013 Microsoft Corporation.

The node replies with a NEGATIVE NAME REGISTRATION RESPONSE because the name is in the Local Name Table and no Conflict Detected flag is set.

4.2 LMHOSTS File Example

The following is an example of an alternate block that may appear in the LMHOSTS file:

```
#BEGIN_ALTERNATE
#INCLUDE  \\Bootsrv3Fileserver\Public\Lmhosts
#INCLUDE  \\Bootsrv4Fileserver\Public\Lmhosts
#INCLUDE  \\Bootsrv9Feilserver\Public\Lmhosts
#END_ALTERNATE
```


Copyright © 2013 Microsoft Corporation.

5 Security Considerations

5.1 Security Considerations for Implementers

The security considerations are unchanged from [RFC1001] and [RFC1002].

5.2 Index of Security Parameters

None.

Copyright © 2013 Microsoft Corporation.

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 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

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: Unless a protocol using the recommended convention specifies otherwise, Windows protocols use the machine system **code page** for NetBIOS names; this means that two computers with different code pages cannot interoperate by using such a protocol with anything other than ASCII names.

<2> Section 2.2.1: All Windows versions do not enforce the asterisk restriction when a name is queried.

<3> Section 2.2.1: Windows 2000, Windows XP, and Windows Server 2003 allow the SMB protocol to register the name "*SMBSERVER".

<4> Section 2.2.3: On Windows, this file is in the systemroot\System32\Drivers\Etc folder.

<5> Section 3.1.1: In Windows NT, Windows 2000, Windows XP, and Windows Server 2003, the default setting is TRUE.

<6> Section 3.1.3: When a DHCP option is received, Windows stores the value in its Node Type, so that the most recent one received is the one used for subsequent operations.

Copyright © 2013 Microsoft Corporation.

<7> Section 3.1.3: On Windows Vista, Windows Server 2008, Windows 7, Windows Server 2008 R2, Windows 8, Windows Server 2012, Windows 8.1, and Windows Server 2012 R2, the read is from the registry using HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\NetBT\Parameters\ReadLMHostsFile if the key exists. A value of 1 means **ReadLMHostsFile** will be set to TRUE. If the key is not present in the registry the value of **ReadLMHostsFile** should not change.

<8> Section 3.1.3: On Windows, because no user name is associated with the computer during startup, NetBT uses a null user name for its credentials when accessing the shared folder where the central LMHOSTS file is located.

<9> Section 3.1.3: To allow null access to a shared folder that contains an LMHOSTS file on a Windows machine, the name of the folder should be set for the registry value of HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet \Services\Lanmanserver \Parameters\NullSessionShares.

<10> Section 3.2.5.1: Windows follows the RFC behavior of replacing addresses if the last byte of the group name is not 0x1c. For group names with the last byte equal to 0x1c, Windows appends addresses. For any group names except those with a last byte of 0x20 or 0x1c, Windows returns 255.255.255.255 in response to queries as if it had stored 255.255.255.255.

Copyright © 2013 Microsoft Corporation.

7 Change Tracking

This section identifies changes that were made to the [MS-NBTE] protocol document between the January 2013 and August 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.

Copyright © 2013 Microsoft Corporation.

- 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
6 Appendix A: Product Behavior	Modified this section to include references to Windows 8.1 operating system and Windows Server 2012 R2 operating system.	Y	Content updated.

Copyright © 2013 Microsoft Corporation.

8 Index

A

Abstract data model
  end node 12
  name server 16
Applicability 8

C

Capability negotiation 8
Change tracking 24

D

Data model - abstract
  end node 12
  name server 16

E

End node
  abstract data model 12
  higher-layer triggered events
    overview 13
    registering NetBios name 13
    resolving NetBios name 14
  initialization 13
  local events 15
  message processing
    handling NAME REGISTRATION REQUEST 14
    overview 14
  sequencing rules
    handling NAME REGISTRATION REQUEST 14
    overview 14
  timer events 15
  timers 12
Examples - overview 18

F

Fields - vendor-extensible 8

G

Glossary 6

H

Higher-layer triggered events
  end node
    overview 13
    registering NetBios name 13
    resolving NetBios name 14
  name server 16

I

Implementer - security considerations 21
Index of security parameters 21
Informative references 7
Initialization
  end node 13
  name server 16
Introduction 6

L

Local events
  end node 15
  name server 17

M

Message processing
  end node
    handling NAME REGISTRATION REQUEST 14
    overview 14
  name server
    handling MULTIHOMED NAME REGISTRATION REQUEST (GROUP) 17
    handling MULTIHOMED NAME REGISTRATION REQUEST (UNIQUE) 17
    handling NAME REGISTRATION REQUEST (GROUP) 16
    overview 16
Messages
  syntax
    MULTIHOMED NAME REGISTRATION REQUEST 9
    NetBIOS name syntax 9
  transport 9

N

Name server
  abstract data model 16
  higher-layer triggered events 16
  initialization 16
  local events 17
  message processing
    handling MULTIHOMED NAME REGISTRATION REQUEST (GROUP) 17
    handling MULTIHOMED NAME REGISTRATION REQUEST (UNIQUE) 17
    handling NAME REGISTRATION REQUEST (GROUP) 16
    overview 16
  sequencing rules
    handling MULTIHOMED NAME REGISTRATION REQUEST (GROUP) 17
    handling MULTIHOMED NAME REGISTRATION REQUEST (UNIQUE) 17
    handling NAME REGISTRATION REQUEST (GROUP) 16
    overview 16
  timer events 17

Copyright © 2013 Microsoft Corporation.

timers 16
NetBIOS name syntax 9
Normative references 7

O

Overview (synopsis) 7

P

Parameters - security index 21
Preconditions 8
Prerequisites 8
Product behavior 22

R

References
informative 7
normative 7
Registering NetBios name 13
Relationship to other protocols 8
Resolving NetBios name
NetBIOS name server selection 14
overview 14

S

Security
implementer considerations 21
parameter index 21
Sequencing rules
end node
handling NAME REGISTRATION REQUEST 14
overview 14
name server
handling MULTIHOMED NAME REGISTRATION REQUEST (GROUP) 17
handling MULTIHOMED NAME REGISTRATION REQUEST (UNIQUE) 17
handling NAME REGISTRATION REQUEST (GROUP) 16
overview 16
Server
abstract data model 16
higher-layer triggered events 16
initialization 16
local events 17
message processing
handling MULTIHOMED NAME REGISTRATION REQUEST (GROUP) 17
handling MULTIHOMED NAME REGISTRATION REQUEST (UNIQUE) 17
handling NAME REGISTRATION REQUEST (GROUP) 16
overview 16
sequencing rules
handling MULTIHOMED NAME REGISTRATION REQUEST (GROUP) 17
handling MULTIHOMED NAME REGISTRATION REQUEST (UNIQUE) 17
handling NAME REGISTRATION REQUEST (GROUP) 16
overview 16
timer events 17
timers 16
Standards assignments 8
Syntax
MULTIHOMED NAME REGISTRATION REQUEST 9
NetBIOS name syntax 9

T

Timer events
end node 15
name server 17
Timers
end node 12
name server 16
Tracking changes 24
Transport 9
Triggered events - higher-layer
end node
overview 13
registering NetBios name 13
resolving NetBios name 14
name server 16

V

Vendor-extensible fields 8
Versioning 8

Copyright © 2013 Microsoft Corporation.