[MC-DPLNAT]:
DirectPlay 8 Protocol:
NAT Locator Specification

Revision Summary

Date	Revision History	Revision Class	Comments
08/10/2007	0.1	Major	Initial Availability
09/28/2007	0.2	Minor	Updated the technical content.
10/23/2007	0.2.1	Editorial	Revised and edited the technical content.
11/30/2007	1.0	Major	Updated and revised the technical content.

Date	Revision History	Revision Class	Comments
01/25/2008	1.0.1	Editorial	Revised and edited the technical content.

Table of Contents

1 Introduction

This document specifies the DirectPlay 8 Protocol: NAT Locator.

The DirectPlay 8 Protocol: NAT Locator Specification are extensions to the DirectPlay 8 Core and Service Providers Protocol (as specified in [MC-DPL8CS]) to improve **Network Address Translation (NAT)** support.

1.1 Glossary

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

Globally Unique Identifier (GUID)
Little-Endian

The following terms are specific to this document:

DirectPlay: A network communication library included with the Microsoft **DirectX** application programming interfaces. **DirectPlay** is a high-level software interface between applications and communication services that makes it easy to connect games over the Internet, a modem link, or a network.

DirectX: Microsoft **DirectX** is a collection of application programming interfaces for handling tasks related to multimedia, especially game programming and video, on Microsoft platforms.

DPNID: A 32-bit identification value assigned to a player as part of its participation in a DirectPlay 8 Core and Service Providers Protocol Session (as specified in [MC-DPL8CS]).

Host: A **DirectPlay host** refers to the DirectPlay 8 Core and Service Providers Protocol (as specified in [MC-DPL8CS]) **session** participant that is performing **session** management duties. All other participants in the **session** are called **peers**.

Network Address Translation (NAT): The process of converting between IP addresses used within an intranet, or other private network, and Internet IP addresses.

Peer: A **DirectPlay peer** refers to a DirectPlay 8 Core and Service Providers Protocol (as specified in [MC-DPL8CS]) **session** participant that is not performing **session** management duties. The participant that is managing the **session** is called the **host**.

Private Address: An IPv4 address that is not globally routable, but is part of the **private address** space specified in [RFC1918] section 3.

Public Address: An external global address used by a **NAT**.

Session: A **DirectPlay session** refers to the collection of participants using the DirectPlay 8 Core and Service Providers Protocol (as specified in [MC-DPL8CS]) to communicate.

User Datagram Protocol (UDP): A common connectionless, datagram-oriented protocol used with the Internet Protocol (IP).

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

1.2 References

This section contains the following information:

Normative References (section 1.2.1) specify stable, published documents that must be read to understand or implement the technology in this document, or whose technology must be present for the technology in this protocol to work. This includes public specifications that define the relevant protocols, and documents that describe the Windows behavior (if other than the protocol specification).

Informative References (section 1.2.2) are published documents that provide additional, optional information relevant to the protocol. For example, an informative reference might provide background or historical information. Informative references are not required to implement the technology in this 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.

[FIPS180] Federal Information Processing Standards Publication, "Secure Hash Standard", FIPS PUB 180-1, April 1995, http://www.itl.nist.gov/fipspubs/fip180-1.htm

[MC-DPL8CS] Microsoft Corporation, "DirectPlay 8 Protocol: Core and Service Providers Specification" September 2007.

[MS-GLOS] Microsoft Corporation, "Windows Protocols Master Glossary", March 2007.

[RFC1918] Rekhter, Y., Moskowitz, B., Karrenberg, D., de Groot, G., and Lear, E., "Address Allocation for Private Internets", RFC 1918, February 1996, http://www.ietf.org/rfc/rfc1918.txt

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

1.2.2 Informative References

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

[RFC3022] Srisuresh, P., and Egevang, K., "Traditional IP Network Address Translator (Traditional NAT)", RFC 3022, January 2001, http://www.ietf.org/rfc/rfc3022.txt

[RFC4380] Huitema, C., "Teredo: Tunneling IPv6 over UDP through Network Address Translations (NATs)", RFC 4380, February 2006, http://www.ietf.org/rfc/rfc4380.txt

[UPNPWANIP] UPnP Forum, "Internet Gateway Device (IGD) V 1.0", November 2001, http://www.upnp.org/standardizeddcps/igd.asp

1.3 Protocol Overview (Synopsis)

The DirectPlay 8 Protocol: NAT Location consists of three separate packet types: path tests, NAT resolver queries, and NAT resolver responses. These optional messages are used to modify behavior of the DirectPlay 8 Core and Service Providers Protocol (as specified in [MC-DPL8CS]) connection process so as to increase support for network environments that involve Network Address Translation (NAT). They are not required for operation of the DirectPlay 8 Core and Service Providers Protocol.

Path tests are packets used to augment the DirectPlay 8 Protocol: Core and Service Providers connection process. While an existing participant is initiating a connection to a new player in response to a DN_MSG_INTERNAL_INSTRUCT_CONNECT message from the **host**, it may configure itself to accept PATH_TEST messages from the new player. Similarly, the new player may begin to periodically send PATH_TEST messages to the existing players from which it expects to receive connection attempts. These PATH_TEST messages are used to create port mappings in NAT or firewall devices that would otherwise prevent the DirectPlay 8 Protocol: Core and Service Providers connection from succeeding.

NAT Resolver Queries and Responses are part of an out-of-band mechanism to enable DirectPlay 8 Protocol: Core and Service Providers hosts to acquire additional addressing information that they can provide to potential clients to improve connectivity in specific, limited NAT scenarios. They enable hosts to create port mappings in NAT or firewall devices, and identify the resulting **public address** and port. This public address and port can then be advertised instead of the local, **private address** and port that hosts normally would advertise.

1.4 Relationship to Other Protocols

These protocol extensions depend on the **User Datagram Protocol (UDP)** and Internet Protocol version 4 (IPv4). The extensions are implemented in conjunction with the DirectPlay 8 Protocol: Core and Service Providers, and nominally the DirectPlay 8 Protocol: Reliable and the DirectPlay 8 Protocol: Host and Port Enumeration. No protocols depend on the extensions' presence. It is not recommended that Network Address Translation (NAT) Resolver queries be performed when a Universal Plug-and-Play (UPnP) Internet Gateway Device (IGD) is configured with a port mapping; instead, the UPnP port mapping should take precedence.

1.5 Prerequisites/Preconditions

The Path Test protocol extension assumes that a DirectPlay 8 Protocol: Core and Service Providers **session** has been established, and that a **peer** is attempting to join a session with the host and at least one other existing peer.

The Network Address Translation (NAT) Resolver Query/Response client/server transaction requires that the responding server be configured with a public or global Internet address. It must not be behind any devices that perform NAT so that it can respond properly to queries from systems that are. It can be performed at any time, but typically occurs when a DirectPlay 8 Protocol: Core and Service Providers host has started.

1.6 Applicability Statement

DirectPlay is designed for multiplayer gaming scenarios. These extensions may be used when additional Network Address Translation (NAT) traversal support is desired for a DirectPlay 8 Protocol: Core and Service Providers gaming session.

These extensions are used when running over Internet Protocol version 4. They are not to be implemented using IPv6. Instead, mechanisms such as the Teredo tunneling specification should address NAT traversal more generically under that protocol.

1.7 Versioning and Capability Negotiation

These protocol extensions have no versioning or capability negotiation.

1.8 Vendor-Extensible Fields

There are no vendor-extensible fields in this protocol.

1.9 Standards Assignments

There are no standards assignments for this protocol.

2 Messages

The following sections specify how DirectPlay 8 Protocol: NAT Locator Specification messages are transported and DirectPlay 8 Protocol: NAT Locator Specification message syntax.

2.1 Transport

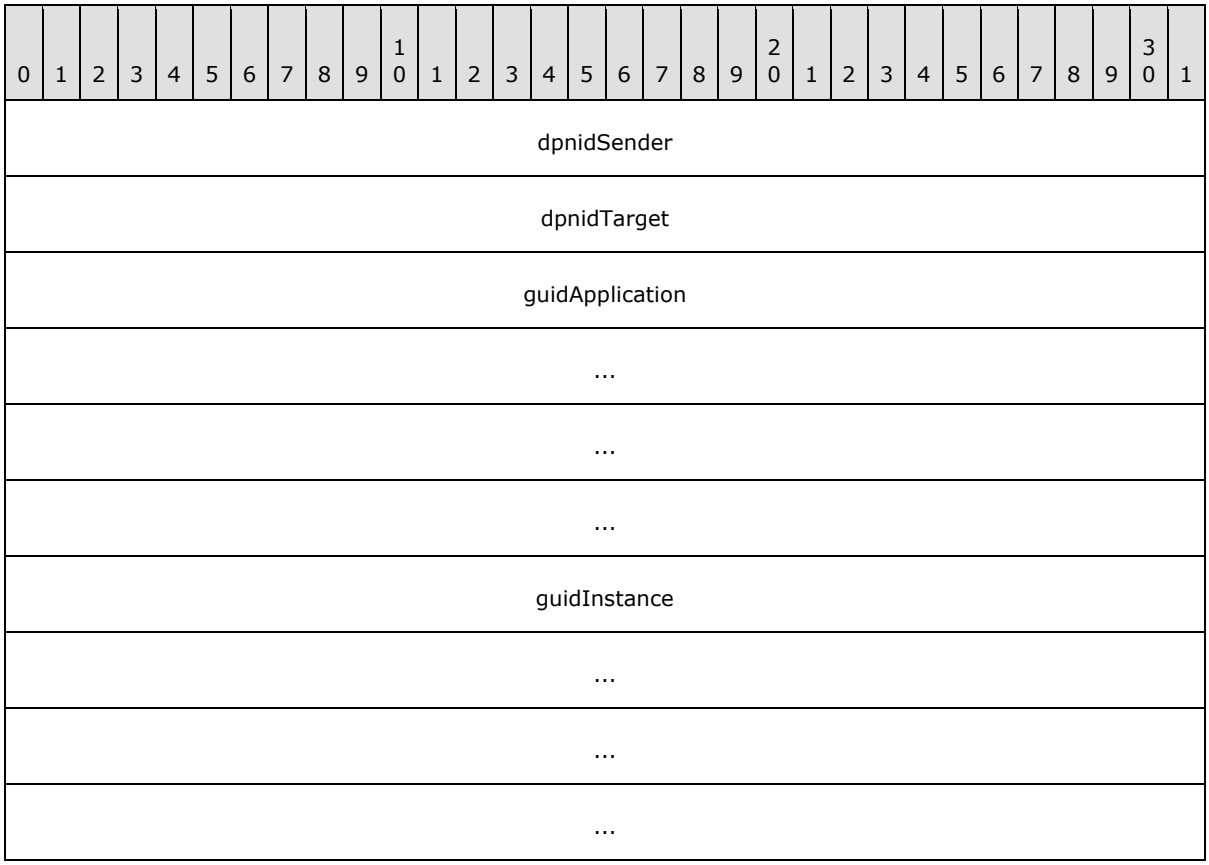
DirectPlay 8 Protocol: NAT Location messages MUST be transported by using UDP. The source and destination port numbers are application-specific and MAY be any value.

2.2 Message Syntax

This section describes the format of messages and pseudo-structures used in the DirectPlay 8 Protocol: NAT Location.

2.2.1 PATHTESTKEYDATA

The PATHTESTKEYDATA is a pseudo-structure that is hashed to generate 64-bit key values.



dpnidSender (4 bytes): The 32-bit **DPNID** value identifying the sending player, in **little-endian** byte order.

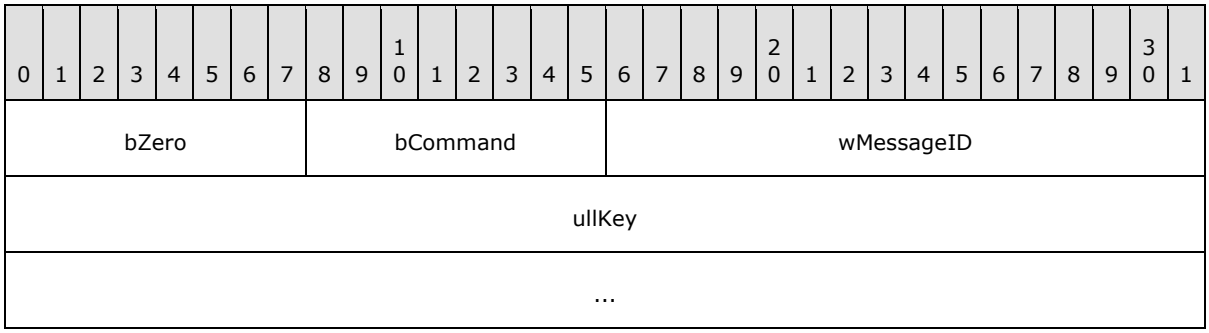
dpnidTarget (4 bytes): The 32-bit DPNID value identifying the intended recipient player, in little-endian byte order.

guidApplication (16 bytes): The 128-bit **GUID** value identifying the DirectPlay 8 Protocol: Core and Service Providers application.

guidInstance (16 bytes): The 128-bit GUID value identifying the particular DirectPlay 8 Protocol: Core and Service Providers session instance.

2.2.2 PATH_TEST

The PATH_TEST messages are sent to trigger outbound Network Address Translation (NAT) and firewall mappings.



bZero (1 byte): An 8-bit identifier used to distinguish this message from DirectPlay 8 Protocol: Reliable messages to the same UDP port. It MUST be set to zero.

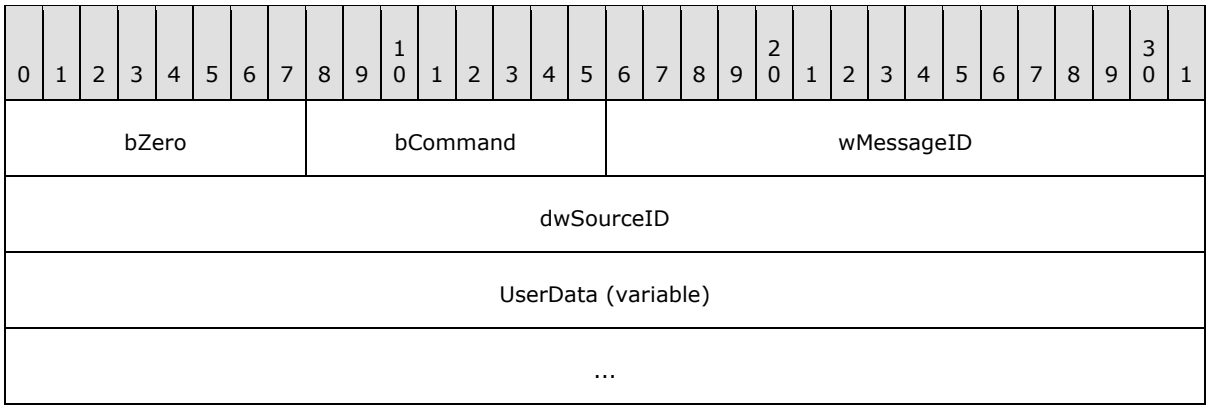
bCommand (1 byte): An 8-bit command code identifying this message as a path test message. It MUST be set to 0x05, PATH_TEST_DATA_KIND (Path Test message type).

wMessageID (2 bytes): A 16-bit value used to uniquely identify an individual PATH_TEST message. This MAY be any value desired by the sender, and MUST be ignored by the receiver. It SHOULD change each time a PATH_TEST message is retried.

ullKey (8 bytes): A 64-bit digest value used to validate the PATH_TEST message. This MUST be generated by using the procedure outlined in section 3.1.3, and MUST be validated by the receiver prior to acting on it.

2.2.3 NAT_RESOLVER_QUERY

The NAT_RESOLVER_QUERY is sent to retrieve translated address information.



bZero (1 byte): An 8-bit identifier used to distinguish this message from DirectPlay 8 Protocol: Reliable messages to the same UDP port. It MUST be set to zero.

bCommand (1 byte): An 8-bit command code identifying this message as a Network Address Translation (NAT) Resolver Query message. It MUST be set to 0x06, NAT_RESOLVER_QUERY_DATA_KIND (NAT Resolver Query message type).

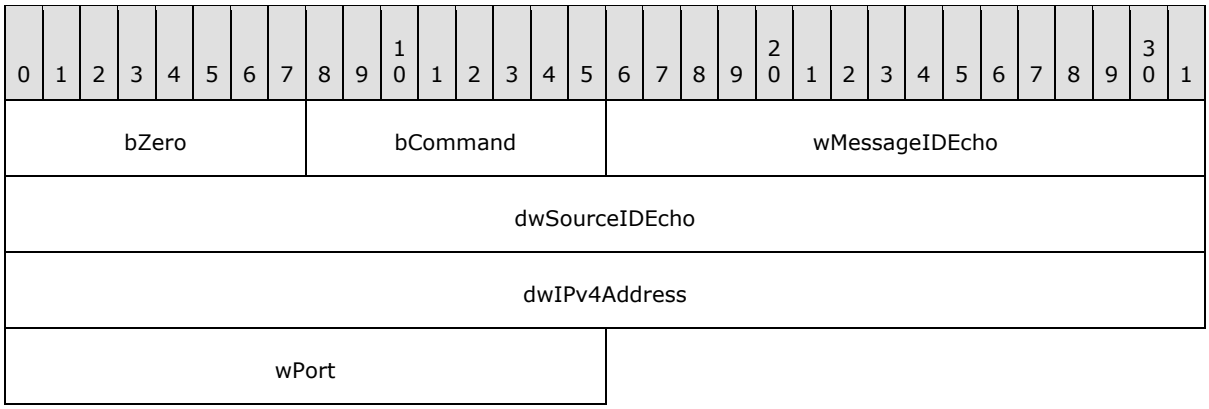
wMessageID (2 bytes): A 16-bit value used by the sender to uniquely identify an individual NAT resolver query message. This MAY be any value desired by the sender, and MUST be echoed in the response message by the receiver. It SHOULD change each time a query message is retried.

dwSourceID (4 bytes): A 32-bit value used by the sender to identify the source of a NAT resolver query message. This MAY be any value desired by the sender, and MUST be echoed in the response message by the receiver.

UserData (variable): An optional, variable length field containing an application-specific query payload. The size is determined by the remaining size of the UDP packet, and MAY be omitted. The receiving application MAY require queries to contain specific **UserData** values in order to respond, or it MAY ignore this field and send replies to all queries.

2.2.4 NAT_RESOLVER_RESPONSE

The NAT_RESOLVER_RESPONSE is sent to report translated address information to the sender of a previous query.



bZero (1 byte): An 8-bit identifier used to distinguish this message from DirectPlay 8 Protocol: Reliable messages to the same UDP port. It MUST be set to zero.

bCommand (1 byte): An 8-bit command code identifying this message as a Network Address Translation (NAT) Resolver Response message. It MUST be set to 0x07, NAT_RESOLVER_RESPONSE_DATA_KIND (NAT Resolver Response message type).

wMessageIDEcho (2 bytes): A 16-bit value used to uniquely identify a response to an individual NAT resolver query message. This MUST be set to the value of **wMessageID** in the query to which this message is a response.

dwSourceIDEcho (4 bytes): A 32-bit value used by the sender to identify the original source to which the NAT resolver response is replying. This MUST be set to the value of **dwSourceID** in the query to which this message is a response.

dwIPv4Address (4 bytes): A 32-bit value indicating the public IPv4 address of the sender of the NAT resolver query. This value MUST be set to the source IPv4 address of the query UDP packet, in network byte order, modified by using the exclusive-or (XOR) bitwise operation with the value of the **dwSourceID** query field.

wPort (2 bytes): A 16-bit value indicating the public port number of the sender of the NAT resolver query. This value MUST be set to the source port number of the query UDP packet, in network byte order, modified by using the exclusive-or (XOR) bitwise operation with the value of the **wMessageID** query field.

3 Protocol Details

The following sections specify details of the DirectPlay 8 Protocol: NAT Locator Specification, including abstract data models, interface method syntax, and message processing rules.

3.1 Path Test 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.

dpnidSender: The 32-bit DPNID value identifying the new Path Test-sending player, in little-endian byte order.

dpnidTarget: The 32-bit DPNID value identifying the existing player, in little-endian byte order.

guidApplication: The 128-bit GUID value identifying the DirectPlay 8 Protocol: Core and Service Providers application.

guidInstance: The 128-bit GUID value identifying the particular DirectPlay 8 Protocol: Core and Service Providers session instance.

ullKey: The 64-bit identification value used to correlate PATH_TEST messages and DirectPlay 8 Protocol: Core and Service Providers connect attempts.

3.1.2 Timers

Retry Timer: Timer used to periodically resend PATH_TEST messages to compensate for potential packet loss. The default settings SHOULD be to retry at intervals of 375 milliseconds with a maximum of 7 attempts, but MAY be any values desired for the particular application and network circumstances.

3.1.3 Initialization

The Path Test protocol extension SHOULD be initialized whenever the DirectPlay 8 Protocol: Core and Service Providers begins connecting an existing peer to a new peer that is attempting to join the session.

To use Path Tests, the peers MUST fill in a PATHTESTKEYDATA pseudo-structure with:

dpnidSender: Set to the DPNID of the new peer, in little-endian byte order.

dpnidTarget: Set to its own DPNID, in little-endian byte order.

guidApplication: Set to the DirectPlay 8 Protocol: Core and Service Providers application GUID.

guidInstance: Set to the DirectPlay 8 Protocol: Core and Service Providers session instance GUID.

They MUST then generate a SHA-1 digest, as specified in [FIPS180], of the PATHTESTKEYDATA binary data, and use the first 64 bits of the output value as the Path Test key value **ullKey**.

For the existing peer, this value MUST remain associated with the connection attempt it is performing until either the attempt fails, a valid reply packet is received from the target address, or it receives a valid PATH_TEST message as described in section 3.1.5.

For the new peer, this value MUST be used in the periodic transmission of PATH_TEST messages as described in section 3.1.6. The Path Test **Retry Timer** MUST be initialized at this time.

3.1.4 Higher-Layer Triggered Events

The DirectPlay 8 Protocol: Core and Service Providers SHOULD inform the Path Test Extension when the connection attempt has completed, whether it was successful or not. The existing peer MUST stop listening for PATH_TEST messages, and the new peer MUST stop transmitting PATH_TEST messages at that time.

3.1.5 Message Processing Events and Sequencing Rules

When the existing peer receives a valid PATH_TEST message, it MUST look for an existing connect attempt that has a matching **ullKey** value. If found, and the connect attempt has not yet received any packets from the intended target IPv4 address and port, the connect target SHOULD be modified to be the source IPv4 address and port of the PATH_TEST message.

If no connect attempt is associated with a matching **ullKey** value, or a matching connect attempt already has received one or more packets, the existing peer MUST ignore the PATH_TEST message.

If the new peer or host receives a PATH_TEST message, it MUST be silently ignored.

3.1.6 Timer Events

Retry Timer: When this timer elapses, the new peer MUST send a new PATH_TEST message to the IPv4 address and port of the existing peer for which it is expecting a connection. This message MUST be sent from the same UDP port number on which it is expecting the connection. If fewer than the maximum number of attempts have been made, the timer MUST then be rescheduled so that it MAY elapse again. Otherwise, the retries have been exhausted and the Path Test operation SHOULD be cancelled.

3.1.7 Other Local Events

No additional local events modify protocol behavior.

3.2 NAT Resolver Response Server Details

3.2.1 Abstract Data Model

No state is required.

3.2.2 Timers

No timers are used.

3.2.3 Initialization

No initialization is necessary.

3.2.4 Higher-Layer Triggered Events

Network Address Translation (NAT) Resolver Response Servers are not required to interact with higher-layers beyond initializing and shutting down.

3.2.5 Message Processing Events and Sequencing Rules

When a Network Address Translation (NAT) Resolver Response server receives a NAT_RESOLVER_QUERY message, it MAY validate the optional **UserData** field by using application-specific logic. If the query is acceptable, the NAT Resolver MUST respond with a NAT_RESOLVER_RESPONSE message to the sender's target IPv4 address and port. The response MUST contain the query's **wMessageID** and **dwSourceID** fields echoed in the corresponding **wMessageIDEcho** and **dwSourceIDEcho** fields. It MUST also set the source IPv4 address of the query in the dwIPv4Address field, in network byte order, and the source port number of the query in the wPort field. Both of these fields MUST also be modified by using **dwSourceID** and **wMessageID**, respectively, with the exclusive-or (XOR) bitwise operation.

If the server receives a NAT_RESOLVER_RESPONSE message, it MUST be silently ignored.

3.2.6 Timer Events

No timers are used.

3.2.7 Other Local Events

No additional local events modify protocol behavior.

3.3 NAT Resolver Query Client Details

3.3.1 Abstract Data Model

No state is required.

3.3.2 Timers

Retry Timer: Timer used to periodically resend NAT_RESOLVER_QUERY messages to compensate for potential packet loss. The default settings SHOULD be to retry at intervals of one second with a maximum of four attempts, but MAY be any values desired for the particular application and network circumstances.

3.3.3 Initialization

The Network Address Translation (NAT) Resolver Query client MAY be initialized whenever the DirectPlay 8 Protocol: Core and Service Providers begins hosting a new session. Initialization consists of scheduling the **Retry Timer**.

3.3.4 Higher-Layer Triggered Events

The DirectPlay 8 Protocol: Core and Service Providers SHOULD inform the Network Address Translation (NAT) Resolver Query client when it is no longer the host of a session. The client MUST abort any query attempts in progress at that time.

3.3.5 Message Processing Events and Sequencing Rules

When a client receives a NAT_RESOLVER_RESPONSE message, it SHOULD validate that **wMessageIDEcho** and **dwSourceIDEcho** correspond to values that it sent in a previous NAT_RESOLVER_QUERY message's **wMessageID** and **dwSourceID** fields. If not, the packet MUST be silently ignored. Otherwise, the client SHOULD reverse the exclusive-or (XOR) bitwise operation performed on the **dwIPv4Address** and **wPort** fields, and save the resulting address and port information for use in advertising the DirectPlay 8 Protocol: Core and Service Providers session. The **Retry Timer** SHOULD then be canceled.

If a client receives a NAT_RESOLVER_QUERY message, it MUST be silently ignored.

3.3.6 Timer Events

Retry Timer: When this timer elapses, the client MUST send a new NAT_RESOLVER_QUERY message to the server. This message MUST be sent from the same UDP port number on which it is the host of the DirectPlay 8 Protocol: Core and Service Providers session. If fewer than the maximum number of attempts have been made, the timer MUST then be rescheduled so that it MAY elapse again. Otherwise, the retries have been exhausted and the Network Address Translation (NAT) Resolver Query operation SHOULD be cancelled.

3.3.7 Other Local Events

No additional local events modify protocol behavior.

4 Protocol Examples

The following section describes a common scenario in which the DirectPlay 8 Protocol: NAT Locator Specification protocol is used.

4.1 Example NAT Resolver Query and Response

Client sends a NAT_RESOLVER_QUERY by using message ID 0x0011 and source ID 0xD4916B52, in little-endian byte order (UDP payload from private address 192.168.0.2:2302 to server at 123.123.123.123:12345):

```
00 06 11 00 52 6B 91 D3
```

Network Address Translation (NAT) device remaps packet source from private address 192.168.0.2:2302 to public address 44.44.44.44:1025.

UDP NAT_RESOLVER_QUERY payload remains the same.

Server sends a NAT_RESOLVER_QUERY echoing message ID 0x0011 and source ID 0xD4916B52, in little-endian byte order. The NAT resolver response server reports the 44.44.44.44:1025 source address, XORed with the source and message IDs (UDP payload from server 123.123.123.123:12345 to public address 44.44.44.44:1025):

```
00 07 11 00 52 6B 91 D3 7E 47 BD F8 15 01
```

5 Security

The following sections specify security considerations for implementers of the DirectPlay 8 Protocol: NAT Locator Specification.

5.1 Security Considerations for Implementers

This protocol uses the SHA-1 hashing algorithm, which has been shown to have weaknesses (as specified in [FIPS180]). However, the protocol is not intended for use in applications that demand robust security without IPSec or other lower-level security mechanisms already in place.

5.2 Index of Security Parameters

Security parameter	Section
SHA-1 digest	3.1.3

6 Appendix A: Windows Behavior

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

- Windows Vista

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

7 Index

S

T

V

W