[MC-NPR]:
.NET Packet Routing 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
08/10/2007	0.1	Major	Initial Availability
09/28/2007	0.2	Minor	Updated the technical content.
10/23/2007	0.3	Minor	Updated the technical content.
11/30/2007	0.3.1	Editorial	Revised and edited the technical content.
01/25/2008	0.3.2	Editorial	Revised and edited the technical content.
03/14/2008	0.3.3	Editorial	Revised and edited the technical content.
05/16/2008	0.3.4	Editorial	Revised and edited the technical content.
06/20/2008	0.4	Minor	Updated the technical content.
07/25/2008	0.5	Minor	Updated the technical content.
08/29/2008	0.5.1	Editorial	Updated index.
10/24/2008	0.5.2	Editorial	Revised and edited the technical content.
12/05/2008	0.5.3	Editorial	Revised and edited the technical content.
01/16/2009	0.5.4	Editorial	Revised and edited the technical content.
02/27/2009	0.5.5	Editorial	Revised and edited the technical content.
04/10/2009	0.5.6	Editorial	Revised and edited the technical content.
05/22/2009	0.5.7	Editorial	Revised and edited the technical content.
07/02/2009	0.5.8	Editorial	Revised and edited the technical content.
08/14/2009	0.5.9	Editorial	Revised and edited the technical content.
09/25/2009	0.6	Minor	Updated the technical content.
11/06/2009	0.6.1	Editorial	Revised and edited the technical content.
12/18/2009	0.6.2	Editorial	Revised and edited the technical content.
01/29/2010	0.7	Minor	Updated the technical content.
03/12/2010	0.7.1	Editorial	Revised and edited the technical content.
04/23/2010	0.7.2	Editorial	Revised and edited the technical content.
06/04/2010	1.0	Major	Updated and revised the technical content.
07/16/2010	2.0	Major	Significantly changed the technical content.

Date	Revision History	Revision Class	Comments
08/27/2010	2.0	No change	No changes to the meaning, language, or formatting of the technical content.
10/08/2010	2.0	No change	No changes to the meaning, language, or formatting of the technical content.
11/19/2010	3.0	Major	Significantly changed the technical content.
01/07/2011	3.0	No change	No changes to the meaning, language, or formatting of the technical content.
02/11/2011	3.0	No change	No changes to the meaning, language, or formatting of the technical content.
03/25/2011	3.0	No change	No changes to the meaning, language, or formatting of the technical content.
05/06/2011	3.0	No change	No changes to the meaning, language, or formatting of the technical content.
06/17/2011	3.1	Minor	Clarified the meaning of the technical content.
09/23/2011	3.1	No change	No changes to the meaning, language, or formatting of the technical content.
12/16/2011	4.0	Major	Significantly changed the technical content.
03/30/2012	4.0	No change	No changes to the meaning, language, or formatting of the technical content.
07/12/2012	4.0	No change	No changes to the meaning, language, or formatting of the technical content.
10/25/2012	4.0	No change	No changes to the meaning, language, or formatting of the technical content.
01/31/2013	4.0	No change	No changes to the meaning, language, or formatting of the technical content.
08/08/2013	4.0	No change	No changes to the meaning, language, or formatting of the technical content.

Contents

Release: Monday, July 22, 2013

1 Introduction

This document specifies the .NET Packet Routing Protocol, which defines a **SOAP header** for indicating that a **SOAP message** can safely be treated as a packet or datagram. A .NET Packet Routing Protocol **router** may select a more efficient routing algorithm to use for the message based on the presence of the SOAP header. The .NET Packet Routing Protocol does not prescribe any specific algorithm or communications infrastructure for forwarding a packet after it has been received by the router.

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]:

> **endpoint**
> **.NET Framework**
> **SOAP header**
> **SOAP message**
> **SOAP mustUnderstand attribute**

The following terms are specific to this document:

> **node:** An **endpoint** in the computer network that can receive, send, or process a SOAP message, as specified in [SOAP1.2-2/2007].

> **packet routing:** A **routing algorithm** that directs a message from a source to an ultimate destination through intermediate routers without guaranteeing that subsequent messages between the source and destination will take the same path.

> **router:** An **endpoint** that receives messages from a **node** and forwards those messages to the next **node** based on a **routing algorithm**.

> **routing algorithm:** An algorithm used to determine the next destination **node** for a specific message.

> **SOAP packet:** A **SOAP message** that does not have a behavioral dependency on the path taken to deliver the message from the source to the destination.

> **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].

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.

[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

[SOAP1.1] Box, D., Ehnebuske, D., Kakivaya, G., et al., "Simple Object Access Protocol (SOAP) 1.1", May 2000, http://www.w3.org/TR/2000/NOTE-SOAP-20000508/

[SOAP1.2-1/2007] Gudgin, M., Hadley, M., Mendelsohn, N., et al., "SOAP Version 1.2 Part 1: Messaging Framework (Second Edition) ", W3C Recommendation 27, April 2007, http://www.w3.org/TR/2007/REC-soap12-part1-20070427/

[SOAP1.2-2/2007] Gudgin, M., Hadley, M., Mendelsohn, N., et al., "SOAP Version 1.2 Part 2: Adjuncts (Second Edition)", W3C Recommendation, April 2007, http://www.w3.org/TR/2007/REC-soap12-part2-20070427

[W3C-XSD] World Wide Web Consortium, "XML Schema Part 2: Datatypes Second Edition", October 2004, http://www.w3.org/TR/2004/REC-xmlschema-2-20041028

[XML1.0] Bray, T., Paoli, J., Sperberg-McQueen, C.M., and Maler, E., "Extensible Markup Language (XML) 1.0 (Second Edition)", W3C Recommendation, October 2000, http://www.w3.org/TR/2000/REC-xml-20001006

1.2.2   Informative References

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

1.3   Overview

Packet routing is the predominant form of routing on the Internet. Packet routing does not preserve connection-oriented features such as routing paths, message-delivery order, or the presence of a back channel for replies (see the following figure that illustrates Packet routing). Circuit routing does preserve those features but requires more resources to route messages (see the following figure that illustrates circuit routing).

If a router receives an application message over a connection, then, without further context about the application or message, the router may infer that the presence of the connection is significant and use circuit routing. However, the application may not require the features of circuit routing for correct delivery of the application message.

The .NET Packet Routing Protocol enables a SOAP message originator to indicate that a message does not have a behavioral dependency on the path taken to deliver the message from the source to the destination. A .NET Packet Routing Protocol router may make use of the indication when selecting among different routing algorithms to apply to the message. The indication provided by the .NET Packet Routing Protocol conveys routing information that may enable the router to select a more efficient routing algorithm.
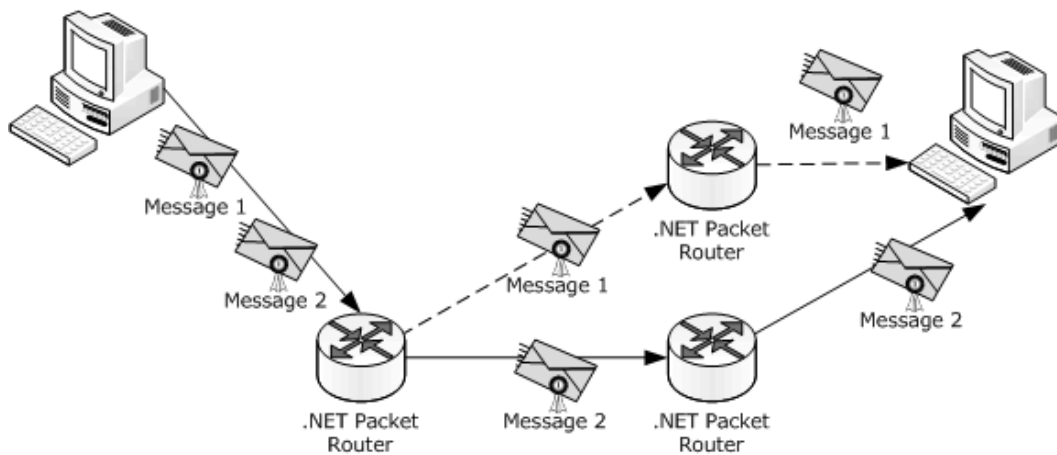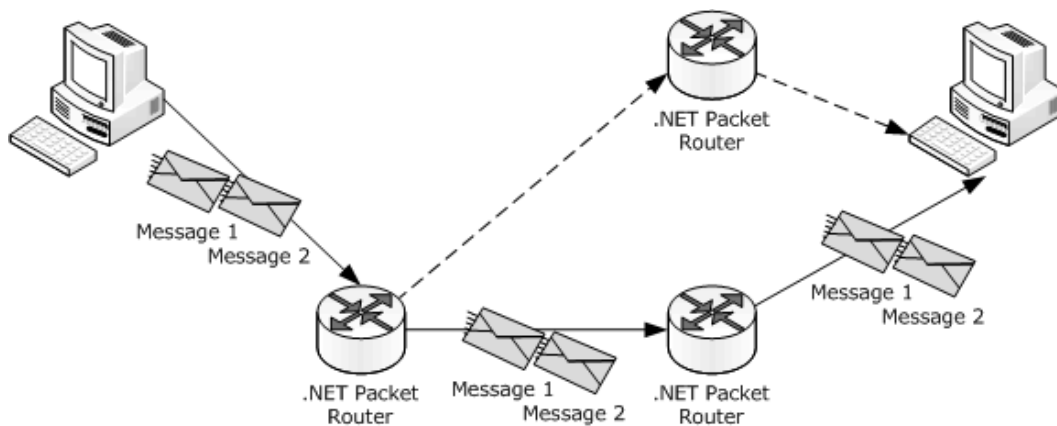
Figure 1: Packet routing of SOAP packets

Figure 2: Circuit routing of SOAP messages

1.3.1 Message Type

For a SOAP message to use packet routing, the message must be a well-formed and complete SOAP message and should have the PacketRoutable element in the SOAP header, as specified in section 2.2.2.

1.4 Relationship to Other Protocols

The .NET Packet Routing Protocol uses any network transport capable of transmitting SOAP 1.1 messages [SOAP1.1] or SOAP 1.2 messages [SOAP1.2-1/2007]. The following figure shows the different layers in a router's stack. At the bottom of the stack is the actual physical communication medium.
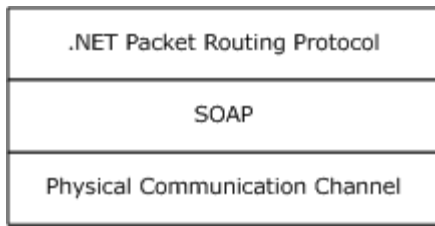
Figure 3: Dependency stack in a SOAP packet router

1.5 Prerequisites/Preconditions

The .NET Packet Routing Protocol requires the communication channel transmitting **SOAP packets** to be capable of producing a self-contained, independent data entity that carries sufficient information to be routed from the source to the destination computer without reliance on earlier exchanges.

1.6 Applicability Statement

The .NET Packet Routing Protocol can be used when an application determines that circuit routing is consuming more resources than necessary. The application requests a downgrade to packet routing by applying a special XML element to the message's SOAP header. For more information, see section 2.1.

1.7 Versioning and Capability Negotiation

This document covers versioning issues in the following areas:

- Protocol versions: This protocol requires the use of SOAP version 1.1 [SOAP1.1] or SOAP version 1.2 [SOAP1.2-1/2007].

- Capability negotiation: The .NET Packet Routing Protocol does not support negotiation of the version to be used.

1.8 Vendor-Extensible Fields

This protocol has no vendor-extensible fields.

1.9 Standards Assignments

None.

2   Messages

2.1   Transport

The .NET Packet Routing Protocol can be used over any transport protocol that supports SOAP version 1.1 [SOAP1.1] or SOAP version 1.2 [SOAP1.2-1/2007]. This specification uses the term SOAP to mean either SOAP 1.1 or SOAP 1.2. Where the differences between the two versions of SOAP are significant, either SOAP 1.1 or SOAP 1.2 is referenced.

Effective use of this protocol requires transmitting any messages that are to be treated as SOAP packets as self-contained, independent data entities that carry sufficient information to be routed from the source to the destination computer without reliance on earlier exchanges. This protocol does not prescribe any specific algorithm or communications infrastructure for forwarding a packet after it has been received by the router.

2.2   Message Syntax

2.2.1   PacketRoutable

PacketRoutable is an XML element [XML1.0] that indicates a node is permitted to treat a SOAP message as a SOAP packet. The PacketRoutable element is specified by the following XML schema [W3C-XSD].

```
<xs:schema
    attributeFormDefault="unqualified"
    elementFormDefault="unqualified"
    targetNamespace="http://schemas.microsoft.com/ws/2005/05/routing"
    xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:element name="PacketRoutable" />
</xs:schema>
```

2.2.2   .NET Packet

A .NET packet is a SOAP packet that contains a PacketRoutable element as a SOAP header. A .NET packet SHOULD NOT include a **SOAP mustUnderstand attribute** for the PacketRoutable element.

3   Protocol Details

The client side of the .NET Packet Routing Protocol MUST implement the insertion of the PacketRoutable element into the SOAP header of the message for a .NET Packet Routing Protocol router to treat it as a packet. Figure 4 represents the state transition of a SOAP message into a .NET packet.
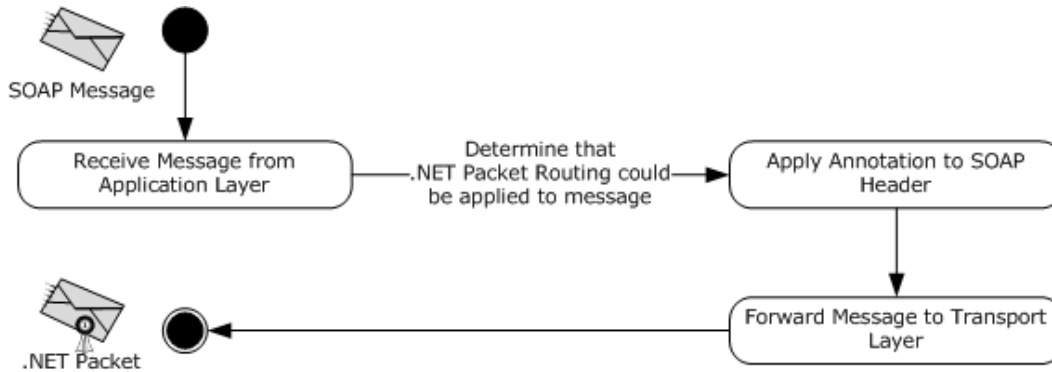

Figure 4: State diagram of client processing of a message

3.1   .NET Packet Router Server Details

Every .NET Packet Routing Protocol router MUST implement processing to receive and send the PacketRoutable element.<1>

When a packet is received by a .NET packet router, and it contains a SOAP header that includes the PacketRoutable element, the router MUST process the XML element and downgrade the routing to packet routing, applying any **routing algorithm** to determine the next destination of the packet. At this stage, an NPR router MAY also process any vendor-specific extension (such as hop count).

When receiving a message from a sending **node** to the NPR router, any failure should be handled based on the specific transport protocol used. For example, if HTTP was selected, an HTTP status code of 200 or 202 may signify that the message was successfully received by the router. An HTTP status code of 500 may signify that the message was not successfully received or processed. The specific handling of errors and .NET Faults is outside the scope of this document.
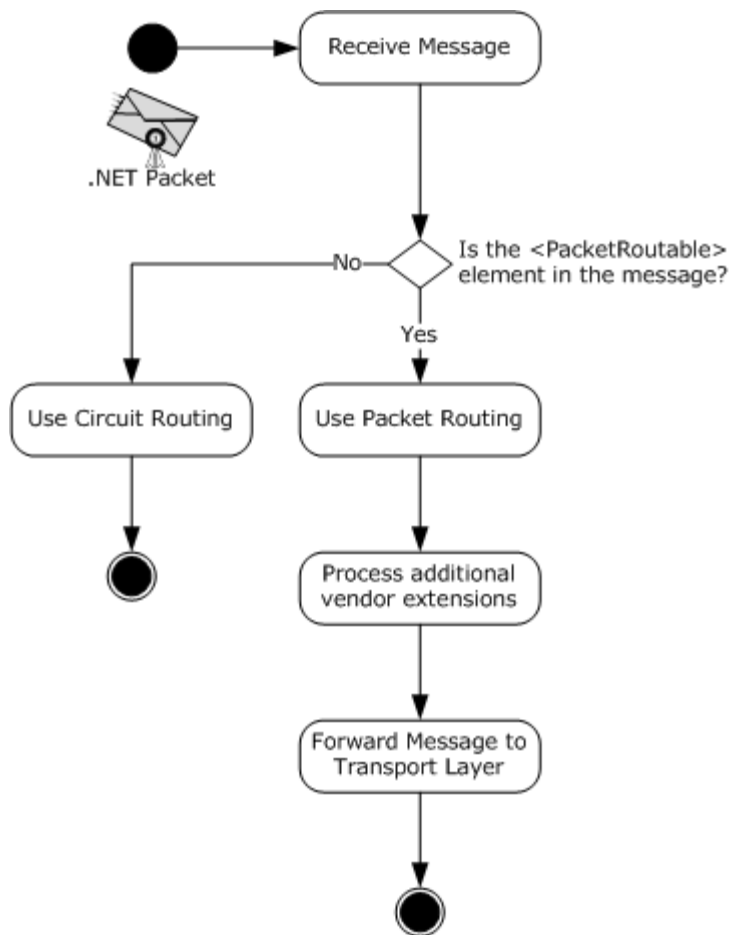
Figure 5: State diagram of the receiving end of a .NET Packet Routing Protocol router

When a .NET Packet Routing Protocol router sends a packet to a destination node, the router MUST maintain the PacketRoutable element in the outgoing packet, and it MAY maintain other vendor-specific extensions. Any outbound transport MAY be chosen based on the available capabilities of either the sending or the receiving router. This protocol does not specify how to determine the appropriate transport protocol or the selection process for the next destination node.

When a .NET Packet Routing Protocol router sends a packet to a destination node, any failure to deliver a message should be handled based on the specific transport protocol that is used. For example, if HTTP was selected, an HTTP status code of 200 or 202 may signify that the message was successfully received by the destination node. An HTTP status code of 500 may signify that the message was not successfully received or processed by the destination node.

Vendor extensions to the protocol MAY implement retry mechanism for resending a message upon failure. This protocol does not recommend any specific handling of such failures.
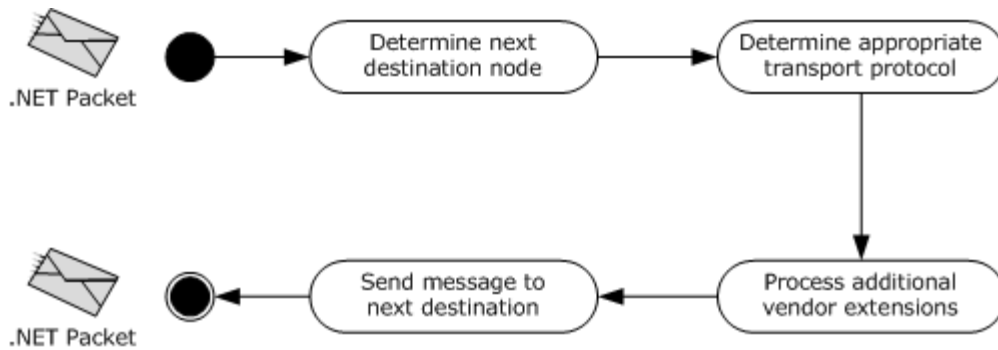
Figure 6: State diagram of the sending side of a .NET Packet Routing Protocol router

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 .NET Packet Routing Protocol does not specify how or when an outbound connection should be opened when a router receives a packet. For example, an implementation of the .NET Packet Routing Protocol MAY require that a packet be fully received before an outbound connection is opened and the packet is forwarded. Another implementation MAY open an outbound connection and start forwarding the data in the packet as it is received on the inbound connection.

3.1.6   Timer Events

None.

3.1.7   Other Local Events

None.

4   Protocol Examples

Messages may have no relation to the underlying transport connection. By inserting the PacketRoutable element into the SOAP header, packet routing would be used instead of circuit routing to deliver the message. Treating the message as a packet enables optimizations such as coalescing messages together or load balancing across multiple back ends to improve reliability, scalability, and performance. The following is an example of a SOAP message that contains the PacketRoutable element in the SOAP header.

```
<s:Envelope xmlns:s="http://www.w3.org/2003/05/soap-envelope"
 xmlns:a="http://www.w3.org/2005/08/addressing">
    <s:Header>
        <PacketRoutable
            xmlns="http://schemas.microsoft.com/ws/2005/05/routing">
        </PacketRoutable>
        <a:Action s:mustUnderstand="1">
            http://tempuri.org/IService/MyOperation
        </a:Action>
        <ActivityId CorrelationId="592d7dfd-aafe-4df8-adfc-003db32efc7
c" xmlns="http://schemas.microsoft.com/2004/09/ServiceModel/Diagnostic
s">9df08e0a-a956-46d9-a859-d6571139ff0f
        </ActivityId>
        <a:To s:mustUnderstand="1">http://localhost:8080/service1
        </a:To>
    </s:Header>
    <s:Body>
        <MyOperation xmlns="http://tempuri.org/">
            <myValue>Some Value</myValue>
        </MyOperation>
    </s:Body>
</s:Envelope>
```

5 Security

5.1 Security Considerations for Implementers

The .NET Packet Routing Protocol does not carry any security consideration. A vendor may extend the protocol to provide additional security considerations as long as the default and basic routing scenarios, as specified by this document, function as expected. In addition, vendors and implementers of this protocol must account for the fallback scenario when a message is routed through a .NET Packet Routing Protocol router that does not support the additional security extensions.

5.2 Index of Security Parameters

None.

6   Appendix A: Product Behavior

This document specifies version-specific details in the Microsoft .NET Framework. For information about which versions of .NET Framework are available in each released Windows product or as supplemental software, see **.NET Framework**.

The information in this specification is applicable to the following Microsoft products or supplemental software. References to product versions include released service packs:

- Microsoft .NET Framework 3.0

- Microsoft .NET Framework 3.5

- Microsoft .NET Framework 4.0

- Microsoft .NET Framework 4.5

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 3.1: The Windows implementation of this specification does not include a .NET Packet Routing Protocol router.

7 Change Tracking

No table of changes is available. The document is either new or has had no changes since its last release.

8   Index