[MS-L2TPIE]: Layer 2 Tunneling Protocol (L2TP) IPsec Extensions

Release: Friday, June 10, 2011

Revision Summary

Date	Revision History	Revision Class	Comments
10/24/2008	0.1		Initial Availability
12/05/2008	0.1.1	Editorial	Editorial Update.
01/16/2009	1.0	Major	Updated and revised the technical content.
02/27/2009	1.1	Minor	Updated the technical content.
04/10/2009	1.2	Minor	Updated the technical content.
05/22/2009	1.3	Minor	Updated the technical content.
07/02/2009	1.4	Minor	Updated the technical content.
08/14/2009	1.4.1	Editorial	Revised and edited the technical content.
09/25/2009	1.5	Minor	Updated the technical content.
11/06/2009	2.0	Major	Updated and revised the technical content.
12/18/2009	2.0.1	Editorial	Revised and edited the technical content.
01/29/2010	2.1	Minor	Updated the technical content.
03/12/2010	2.1.1	Editorial	Revised and edited the technical content.
04/23/2010	2.1.2	Editorial	Revised and edited the technical content.
06/04/2010	2.1.3	Editorial	Revised and edited the technical content.
07/16/2010	2.1.3	No change	No changes to the meaning, language, or formatting of the technical content.
08/27/2010	3.0	Major	Significantly changed the technical content.
10/08/2010	4.0	Major	Significantly changed the technical content.
11/19/2010	4.0	No change	No changes to the meaning, language, or formatting of the technical content.
01/07/2011	4.1	Minor	Clarified the meaning of the technical content.
02/11/2011	4.2	Minor	Clarified the meaning of the technical content.
03/25/2011	4.2	No change	No changes to the meaning, language, or formatting of the technical content.
05/06/2011	4.2	No change	No changes to the meaning, language, or formatting of the technical content.
06/17/2011	4.3	Minor	Clarified the meaning of the technical content.

Contents

1 Introduction

The Layer 2 Tunneling Protocol (L2TP) is an Internet Engineering Task Force (IETF) standard protocol that allows IP, IPX, or NetBEUI traffic to be encrypted, and then sent over any medium that supports point-to-point (PPP) datagram delivery, such as IP, X.25, Frame Relay, or ATM (Point to Point Protocol [RFC1661]). See [RFC2661] section 1 for an introduction to L2TP. [RFC3193] specifies an Internet Engineering Task Force (IETF) standard protocol designed to use Internet Protocol Security (IPsec) [RFC2401] to provide for tunnel authentication, privacy protection, and integrity checking and replay protection of L2TP.

This document specifies a set of vendor-specific options as well as non-standard options for Layer 2 Tunneling Protocol IPsec.

In this document **LAC** (L2TP Access Concentrator) and client are used interchangeably, similarly **LNS** (L2TP Network Server) and server are used interchangeably.

1.1 Glossary

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

AV pair
globally unique identifier (GUID)
session
tunnel

The following terms are specific to this document:

L2TP Access Concentrator (LAC): A node that acts as one side of an L2TP tunnel endpoint and is a peer to the L2TP Network Server (LNS). The LAC sits between an LNS and a remote system and forwards packets to and from each. Packets sent from the LAC to the LNS require tunneling with the L2TP protocol as defined in this document. The connection from the LAC to the remote system is either local or a PPP link.

L2TP Network Server (LNS): A node that acts as one side of an L2TP tunnel endpoint and is a peer to the L2TP Access Concentrator (LAC). The LNS is the logical termination point of a PPP session that is being tunneled from the remote system by the LAC.

peer: When used in context with L2TP, peer refers to either the LAC or LNS. LNS is a peer to LAC and vice versa.

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 Specification documents 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.

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.

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

[IANA-ENT] Internet Assigned Numbers Authority, "Private Enterprise Numbers", January 2007, http://www.iana.org/assignments/enterprise-numbers

[L2TP draft] Townsley, W., Valencia, A., Rubens, A., et al., "Layer Two Tunneling Protocol L2TP", draft-ietf-pppext-l2tp-12.txt, February 1999, http://tools.ietf.org/id/draft-ietf-pppext-l2tp-12.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

[RFC2661] Townsley, W., Valencia, A., Rubens, A., et al., "Layer Two Tunneling Protocol L2TP", RFC 2661, August 1999, http://www.ietf.org/rfc/rfc2661.txt

[RFC3193] Patel, B., Aboba, B., Zorn, G., and Booth, S., "Securing L2TP using IPsec", RFC 3193, November 2001, http://www.ietf.org/rfc/rfc3193.txt

1.2.2 Informative References

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

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

[RFC1661] Simpson, W., Ed., "The Point-to-Point Protocol (PPP)", STD 51, RFC 1661, July 1994, http://www.ietf.org/rfc/rfc1661.txt

[RFC2401] Kent, S., and Atkinson, R., "Security Architecture for the Internet Protocol", RFC 2401, November 1998, http://www.ietf.org/rfc/rfc2401.txt

1.3 Overview

L2TP IPsec Extensions (L2TPIE) provides extensions to L2TP [RFC2661] and to securing L2TP with IPsec [RFC3193] in order to provide traceability and data control flow features. In this extension a new Microsoft vendor-specific **AV pair** is sent in control messages from the client to the server so that tracing events on the server specific to a client can be correlated. This extension uses the data control flow mechanism specified in [L2TP draft].

1.4 Relationship to Other Protocols

This protocol is based on L2TP [RFC2661], [L2TP draft], and securing L2TP with IPsec [RFC3193] protocols. L2TPIE supports only IPsec transport mode. The following network stack diagram demonstrates the relationship of L2TP with other protocols in an IPsec transport mode.

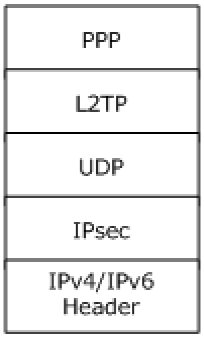

Figure 1: L2TP network stack

1.5 Prerequisites/Preconditions

None beyond those specified in [RFC2661], [L2TP draft], and [RFC3193].

1.6 Applicability Statement

This protocol is applicable when the implementation uses L2TP [RFC2661] and secures L2TP with IPsec [RFC3193].

1.7 Versioning and Capability Negotiation

L2TPIE is based on version 2 of the L2TP protocol, as specified in section 3.1 of [RFC2661].

1.8 Vendor-Extensible Fields

The vendor-extensible fields described in this document comply with section 4.1 of [RFC2661], which specifies how vendor-specific AV pair are passed. The vendor ID for Microsoft vendor-specific AVPs is 0x137. The vendor-extensible options used by L2TP are specified in section 2.2.1.

1.9 Standards Assignments

The only standards assignment required for this protocol is *Private Enterprise Number*. The required value for this parameter is 311 (see [IANA-ENT] for details).

2 Messages

2.1 Transport

All L2TP attributes are transported within L2TP, which is transported over UDP [RFC768] as specified in section 8.1 of [RFC2661] and IPsec, as specified in [RFC3193]. L2TP LNS listens for L2TP messages on the UDP port.<1>

2.2 Message Syntax

These L2TP IPsec extensions use the message format for vendor-specific options, as specified in [RFC2661] section 4.1.

All option fields and values described in this document are sent in network-byte order unless indicated otherwise.

2.2.1 L2TP AV pairs

L2TP AV pairs have Vendor ID fields. The value 0, corresponding to IETF adopted attribute values, is used for all AV pair defined in [RFC2661]. This specification defines the Microsoft vendor-specific AV pair ATTR_VEN_MS_CorrID_Type. The Vendor ID for Microsoft vendor-specific AVPs is 0x137.

Field Name	Value	Meaning
Attribute Type	0x01 (ATTR_VEN_MS_CorrID_Type)	Microsoft vendor-specific correlation ID type (2.2.1.1)

<2>

2.2.1.1 L2TP AV Pair: Microsoft Vendor-specific Correlation ID Type (0x01)

This option is sent in ICRQ control messages so that any tracing application is able to correlate the messages pertaining to a particular tunnel context on the LAC and LNS.

0	1	2	3	4	5	6	7	8	9	10	1	2	3	4	5	6	7	8	9	20	1	2	3	4	5	6	7	8	9	30	1
M	H	rsvd				Length										Vendor ID															
Attribute type																Attribute value															
...																															
...																															
...																															
...																															

M (1 bit): The Mandatory (M) bit. MUST be set to 0.

H (1 bit): The Hidden (H) bit. MUST be set to 0.

rsvd (4 bits): Reserved bits. MUST be set to 0.

Length (10 bits): MUST be set to 22.

Vendor ID (2 bytes): MUST be set to 0x137.

Attribute type (2 bytes): MUST be set to 0x01.

Attribute value (16 bytes): MUST be a 16-byte **GUID**, as specified in [MS-DTYP], (section 2.3.2.2). This SHOULD be unique for every ICRQ request on an LAC/LNS combination.

2.2.2 L2TP Congestion Control (Reset)

L2TPIE implements the flow control for data packets, as defined in section 4.3 of [L2TP draft]. (Flow control was removed from [RFC2661].)

3 Protocol Details

3.1 Common (LAC/LNS) Details

3.1.1 Abstract Data Model

The L2TPIE state machine MUST conform to the state machine specified in section 7 of [RFC2661].

3.1.2 Timers

These extensions do not define any timers beyond those described in [RFC2661] and [RFC3193].

3.1.3 Initialization

These extensions do not define any initialization beyond that specified in [RFC2661] and [RFC3193]. The configuration values of various configurable parameters are read from the registry.<3>

3.1.3.1 Securing L2TP with IPsec

The L2TP per-packet security check is specified in section 3.3 of [RFC3193]. The LAC/LNS MAY verify the UDP port value in the packet that is received with the socket information that is used to set up the L2TP tunnel.

Section 4 of [RFC3193] specifies IPsec filtering details when protecting L2TP.

Section 4.2.3 of [RFC3193] specifies how a Responder chooses a new address, if it is capable of doing so. The responder MUST NOT choose a new IP address during L2TP negotiation. The client SHOULD disconnect the session when it receives a StopCCN message. It SHOULD NOT check the Result code or the error code.

Section 2 of [RFC3193] specifies the L2TP security requirements.

Section 4.2.2 of [RFC3193] specifies filters for protecting L2TP.

Section 5.1.4 of [RFC3193] specifies usage of pre-shared keys.

3.1.4 Higher-Layer Triggered Events

The higher layer can initiate a request to disconnect an established call **session**. When the higher layer requests to disconnect an established call session, the LAC/LNS MUST initiate the session teardown, as specified in section 5.6 of [RFC2661]. After the session teardown is complete, if no more call sessions exist between LAC and LNS, the control connection between LAC and LNS SHOULD be disconnected by initiating the control connection teardown, as specified in section 5.7 of [RFC2661].

3.1.5 Message Processing Events and Sequencing Rules

None beyond those specified in [RFC2661] and [RFC3193].

3.1.5.1 Header Format

All reserved bits MUST be set to 0 on outgoing messages and ignored on incoming messages (section 3.1 of [RFC2661]). If the L2TP header of a received packet has any bits set other than T, L,

and S (the reserved bits as specified in [RFC2661]), the LAC/LNS discards the packet. For outgoing calls, the LNS will stop the session setup attempt and will not resend the OCRQ.<4>

Section 5.8 of [RFC2661] specifies the sequence number in the Header.<5>

3.1.5.2 Control Message AV Pairs

Section 4 of [RFC2661] specifies control message AV pairs.

If a received message has an AVP with

- any of the reserved bits set to 1 or,
- the M bit set and a non-IETF vendor-id value or,
- an unrecognized Attribute value or,
- the M and H bits set, and the AVP has an attribute other than the following:
  - Proxy Authen Name
  - Proxy Authen Challenge
  - Proxy Authen Id
  - Proxy Authen Response
  - Called Number
  - Calling Number
  - Sub-Address
  - Initial Received LCP CONFREQ
  - Last Sent LCP CONFREQ
  - Last Received LCP CONFREQ
  - ACCM
  - Private Group ID

then

- if it is one of the tunnel establishment messages such as SCCRP or SCCRQ, then the message is ignored.
- if it is one of the call establishment messages like ICRQ or SCCCN, then the tunnel is brought down.
- for the rest of the messages the session is torn down.

The following AV pairs are ignored if they are hidden (H bit set to 1), even if the M bit is set.

- Proxy Authen Name
- Proxy Authen Challenge

- Proxy Authen ID
- Proxy Authen Response
- Called Number
- Calling Number
- Sub Address
- Initial Received LCP CONFREQ
- Last Sent LCP CONFREQ
- Last Received LCP CONFREQ:
- ACCM
- Private Group ID

None of the AV pairs sent by LAC/LNS have the H bit set to 1.

Section 4.4 of RFC [RFC2661] provides a list of all L2TP AV pairs, some of which may be Hidden.

3.1.5.3 Start-Control-Connection-Request (SCCRQ)

Section 6.1 of [RFC2661] specifies the AV pairs for the Start-Control-Connection-Request (SCCRQ) message.<6>

3.1.5.4 Start-Control-Connection-Reply (SCCRP)

Start-Control-Connection-Reply (SCCRP) is the control message sent in response to SCCRQ. Section 6.2 of [RFC2661] specifies the AV pairs used with SCCRP.<7>

3.1.5.5 Start-Control-Connection-Connected (SCCCN)

Start-Control-Connection-Connected (SCCCN) is the control message sent in response to SCCRP. Section 6.3 of [RFC2661] specifies the AV pairs used with SCCCN.<8>

3.1.5.6 Stop-Control-Connection-Notification (StopCCN)

Stop-Control-Connection-Notification (StopCCN) is the control message sent to inform the **peer** that the **tunnel** is being shut down and the control connection should be closed. Section 6.4 of [RFC2661] specifies the AV pairs used by StopCCN.<9>

3.1.5.7 Hello (HELLO)

Hello (HELLO) is the control message used as a "keepalive" for the tunnel; see section 6.5 of [RFC2661] for details.<10>

3.1.5.8 Call-Disconnect-Notify (CDN)

Call-Disconnect-Notify (CDN) is the control message sent by an LAC or LNS to request disconnection of a specified call within a tunnel, as specified in section 6.2 of [RFC2661].<11>

3.1.5.9 Set-Link-Info (SLI)

The Set-Link-Info (SLI) message is an L2TP control message sent by the LNS to the LAC to set PPP-negotiated options, as specified in section 6.14 of [RFC2661]. The LAC MUST ignore this message and it SHOULD not be sent by the LNS.

3.1.6 Timer Events

None beyond those specified in [RFC2661] and [RFC3193].

3.1.7 Other Local Events

None.

3.2 LAC/Client Details

3.2.1 Abstract Data Model

The L2TPIE state machine MUST conform to the state machine specified in [RFC2661] section 7.<12>

3.2.2 Timers

No timers are defined beyond those described in [RFC2661] and [RFC3193].

3.2.3 Initialization

None beyond those specified in [RFC2661] and [RFC3193].

3.2.4 Higher-Layer Triggered Events

Apart from the disconnect event specified in section 3.1.4, a LAC can receive a request to establish an L2TP call session from the higher layer. To establish an L2TP call session, the LAC MUST establish a control connection with LNS, as specified in section 5.1 of [RFC2661], if not already established. Once the control connection is available, LAC MUST send the Incoming-Call-Request message (section 6.6 of [RFC2661]), as specified in section 5.2.1 of [RFC2661], to establish the L2TP call session with LNS.

3.2.5 Message Processing Events and Sequencing Rules

None beyond those specified in [RFC2661] and [RFC3193].

3.2.5.1 Incoming-Call-Request (ICRQ)

Incoming Call Request (ICRQ) specifies the control message sent by the LAC to an LNS when an incoming call is detected (see section 6.6 of [RFC2661]).<13>

The Microsoft vendor-specific correlation ID type (section 2.2.1.1) AV pair SHOULD be sent in the ICRQ message and SHOULD also be logged on the client machine for debugging.

3.2.5.2 Incoming-Call-Connected (ICCN)

Incoming-Call-Connected (ICCN) is the control message sent by the LAC to LNS in response to a received ICRP message (see section 6.8 of [RFC2661]). The LAC MAY send a Receive Window Size AVP in the ICCN message.<14>

3.2.5.3 Outgoing-Call-Reply (OCRP)

Outgoing-Call-Reply (OCRP) is the control message sent by the LAC to an LNS in response to an OCRQ (see section 6.10 of [RFC2661]). <15>

3.2.5.4 Outgoing-Call-Connected (OCCN)

Outgoing-Call-Connected (OCCN) is the control message sent by LAC to LNS following the OCRP, and after the outgoing call has been completed (see section 6.11 of [RFC2661]). The LAC MAY send a Receive Window Size AV pair and Sequencing Required AV pair in the OCCN message.<16>

3.2.5.5 WAN-Error-Notify (WEN)

WAN-Error-Notify (WEN) specifies the control message sent by LAC to LNS to indicate a WAN error condition (see section 6.13 of [RFC2661]).<17>

3.2.6 Timer Events

None beyond those specified in [RFC2661] and [RFC3193].

3.2.7 Other Local Events

Tracing or debugging logs on the LAC/Client SHOULD include the ATTR_VEN_MS_CorrID_Type specified in 2.2.1.1. Debugging connection setup issues on the LAC/Client is made easy by looking for the corresponding log on the LNS. Because ATTR_VEN_MS_CorrID_Type is unique for every connection a tracing/debugging application will be able to correlate the events specific to a connection on the LNS.

3.3 LNS/Server Details

3.3.1 Abstract Data Model

The L2TPIE state machine MUST conform to the state machine specified in section 7 of [RFC2661].

3.3.2 Timers

None beyond than those required for implementation as specified in section 5.8 of [RFC2661].

3.3.3 Initialization

None beyond those specified in [RFC2661] and [RFC3193].

3.3.4 Higher-Layer Triggered Events

None beyond the specification in section 3.1.4.

3.3.5 Message Processing Events and Sequencing Rules

None beyond those specified in [RFC2661] and [RFC3193].

3.3.5.1 Incoming-Call-Reply (ICRP)

Incoming-Call-Reply (ICRP) specifies the control message sent by the LNS to an LAC in response to ICRQ (see section 6.7 of [[RFC2661]).<18>

3.3.5.2 Incoming-Call-Connected (ICCN)

Incoming-Call-Connected (ICCN) is the control message sent by the LAC to LNS in response to a received ICRP message (see section 6.8 of [RFC2661]). The LAC MAY send the Receive Window Size AV pair. The LNS MAY close the call if the received Framing Type AV pair has a value other than Synchronous Framing (1) in the ICCN message.<19>

3.3.5.3 Outgoing-Call-Request (OCRQ)

Outgoing-Call-Request (OCRQ) is the control message sent by the LNS to the LAC to indicate that an outbound call from the LAC is to be established (see section 6.9 of [RFC2661]). The LNS SHOULD NOT send the Called Number AV pair in the OCRQ message.<20>

3.3.5.4 Outgoing-Call-Reply (OCRP)

Outgoing-Call-Reply (OCRP) specifies the control message sent by the LAC to the LNS in response to OCRQ (see section 6.10 of [RFC2661]). <21>

3.3.5.5 Outgoing-Call-Connected (OCCN)

Outgoing-Call-Connected (OCCN) specifies the control message sent by an LAC to an LNS following the OCRP and after the outgoing call has been completed (see section 6.11 of [RFC2661]).<22>

3.3.5.6 WAN-Error-Notify (WEN)

WAN-Error-Notify (WEN) specifies the control message sent by an LAC to an LNS to indicate a WAN error condition (see section 6.13 of [RFC2661]).<23>

3.3.6 Timer Events

None beyond those specified in [RFC2661] and [RFC3193].

3.3.7 Other Local Events

Tracing or debugging logs on the LNS/Server SHOULD include the ATTR_VEN_MS_CorrID_Type specified in 2.2.1.1. Debugging connection setup issues on the LAC/Client is made easy by looking for the corresponding log on the LNS/Server. Because ATTR_VEN_MS_CorrID_Type is unique for every connection a tracing/debugging application will be able to correlate the events specific to a connection on the LNS/Server.

4 Protocol Examples

The following example shows the sequence of messages exchanged when a machine running Windows Vista® operating system with Service Pack 1 (SP1) (name: "testclient.contoso.com") with IP address 1.1.1.1(client 100mbps connection) establishes an L2TP tunnel with a machine running Windows Server® 2008 operating system (name: "testserver.contoso.com") with IP address 2.2.2.2(server, 100mbps connection).

Before starting L2TP protocol the following filters are applied:

Server plumbs following IPsec filters during its initialization.

Inbound Filters:

```
Source Address: Any, Destination Address: 2.2.2.2, Protocol: UDP, Source port:1701,
Destination Port: Any

Source Address: Any, Destination Address: 2.2.2.2, Protocol: UDP, Source port:1701,
Destination Port: 1701

Source Address: Any, Destination Address: 2.2.2.2, Protocol: UDP, Source port: Any,
Destination Port: 1701
```

Outbound Filters:

```
Source Address: 2.2.2.2, Destination Address: Any, Protocol: UDP, Source port: Any,
Destination Port: 1701

Source Address: 2.2.2.2, Destination Address: Any, Protocol: UDP, Source port: 1701,
Destination Port: 1701

Source Address: 2.2.2.2, Destination Address: Any, Protocol: UDP, Source port: 1701,
Destination Port: Any
```

Client plumbs following IPsec filters before it connects to VPN server.

Inbound Filters:

```
Source Address: 2.2.2.2, Destination Address: 1.1.1.1, Protocol: UDP,
Source port: Any, Destination Port: 1701


Source Address: 2.2.2.2, Destination Address: 1.1.1.1, Protocol: UDP,
Source port: 1701, Destination Port: 1701
```

Outbound Filters:

```
Source Address: 1.1.1.1, Destination Address: 2.2.2.2, Protocol: UDP,
Source port: 1701, Destination Port: Any
```

```
Source Address: 1.1.1.1, Destination Address: 2.2.2.2, Protocol: UDP,
Source port: 1701, Destination Port: 1701
```

Process:

```
Start-Control-Connection-Request (SCCRQ) is a control message used to
initialize a tunnel between an LNS and an LAC. LAC initiates the tunnel
establishment process in this example. A UDP packet with source IP address
1.1.1.1 and source port 1701 is sent by LAC to destination IP address
2.2.2.2  and destination port 1701 to begin the tunnel establishment process.
Following are the details of the L2TP packets:

    Header:
        T bit is set to 1 to indicate control message
        L bit is set to 1 to indicate length is present
        S bit is set to indicate Nr and Ns are present
        Version field is set to 0x02
        Length field is set to 0x84
        All other fields in the header are set to 0

    The following AVPs are sent:

    Message Type:
        Flags:  M is set to 0x1, length is set to 0x8, rest are 0x0
        VendorId is set to 0x0 (IETF)
        Attribute Type is set to 0x0 (Message type)
        Message Type in Attribute Value is set to 0x0001 (SCCRQ)
        Length is set to 0x8

    Protocol Version:
        Flags:  M is set to 0x1, length is set to 0x8, all other flags are 0x0
        VendorId is set to 0x0 (IETF)
        Attribute Type is set to 0x2 (Protocol Version)
        Ver field is set to 0x01
        Rev field is set to 0x0

    Framing Capabilities:
        Flags:  M is set to 0x1, length is set to 0x10, all other flags are 0x0
        VendorId is set to 0x0 (IETF)
        Attribute Type is set to 0x3 (Framing Capabilities)
        S bit is set to 0x1(Synchronous framing), all other bits are 0x0

    Bearer Capabilities:
        Flags:  length is set to 0x10, all other flags are 0x0
        VendorId is set to 0x0 (IETF)
        Attribute Type is set to 0x4 (Bearer Capabilities)
        all other fields are 0x0

    Firmware revision:
        Flags:  length is set to 0x8, all other flags are 0x0
        VendorId is set to 0x0 (IETF)
        Attribute Type is set to 0x6 (Firmware revision)
        Firmware version is set to (0x600)
```

```
Host name:
    Flags:  M is set to 0x1, length is set to 0x16 rest are 0x0
    VendorId is set to 0x0 (IETF)
    Attribute Type is set to 0x7 (Host name)
    Hostname is set to ASCII string "testclient.contoso.com"

Vendor name:
    Flags: length is set to 0x0F, all other flags are 0x0
    VendorId is set to 0x0 (IETF)
    Attribute Type is set to 0x8 (Vendor name)
    Vendor name is set to ASCII string "Microsoft"

Assigned Tunnel
    Flags:  M is set to 0x1, length is set to 0x8, all other flags are 0x0
    VendorId is set to 0x0 (IETF)
    Attribute Type is set to 0x9 (Assigned Tunnel)
    Assigned tunnelID is set to 0xd (just a sample)

Receive Window
    Flags:  M is set to 0x1, length is set to 0x8, all other flags are 0x0
    VendorId is set to 0x0 (IETF)
    Attribute Type is set to 0xa (Receive Window)
    Window size is set to 0x8

On Receiving the above packet, the server responds with SCCRP with the following details:
    Header:
        T bit is set to 1 to indicate control message
        L bit is set to 1 to indicate length is present
        S bit is set to indicate Nr and Ns are present
        Version field is set to 0x02
        Length field is set to 0x68
        TunnelId is set to 0xD (this value is extracted from Assigned Tunnel
          AVP in SCCRP)
        Sequence Number expected is set to 0x1
        All other fields in the header are set to 0

    The following AVPs are sent:

    Message Type:
        Flags:  M is set to 0x1, length is set to 0x8, all other flags are 0x0
        VendorId is set to 0x0 (IETF)
        Attribute Type is set to 0x0 (Message type)
        Message Type in Attribute Value is set to 0x0002 (SCCRP)
        Length is set to 0x8

    Protocol Version:
        Flags:  M is set to 0x1, length is set to 0x8, all other flags are 0x0
        VendorId is set to 0x0 (IETF)
        Attribute Type is set to 0x2 (Protocol Version)
        Ver field is set to 0x01
        Rev field is set to 0x0

    Framing Capabilities:
        Flags:  M is set to 0x1, length is set to 0x10, all other flags are 0x0
        VendorId is set to 0x0 (IETF)
        Attribute Type is set to 0x3 (Framing Capabilities)
        S bit is set to 0x1(Synchronous framing), all other bits are 0x0

    Bearer Capabilities:
```

```
        Flags:  length is set to 0x10, all other flags are 0x0
        VendorId is set to 0x0 (IETF)
        Attribute Type is set to 0x4 (Bearer Capabilities)
        All other fields are 0x0

    Host name:
        Flags:  M is set to 0x1, length is set to 0x16, all other flags are 0x0
        VendorId is set to 0x0 (IETF)
        Attribute Type is set to 0x7 (Host name)
         Hostname is set to ASCII string "testserver.contoso.com"

    Vendor name:
        Flags: length is set to 0x0F, all other flags are 0x0
        VendorId is set to 0x0 (IETF)
        Attribute Type is set to 0x8 (Vendor name)
        Hostname is set to ASCII string "Microsoft"

    Assigned Tunnel
        Flags:  M is set to 0x1, length is set to 0x8, all other flags are 0x0
        VendorId is set to 0x0 (IETF)
        Attribute Type is set to 0x9 (Assigned Tunnel)
        Assigned tunnelID is set to 0xd

    Receive Window
        Flags:  M is set to 0x1, length is set to 0x8, all other flags are 0x0
        VendorId is set to 0x0 (IETF)
        Attribute Type is set to 0xa (Receive Window)
        Window size is set to 0x8

On receiving the SCCRP the client responds with SCCCN with the following details:

    Header:
        T bit is set to 1 to indicate control message
        L bit is set to 1 to indicate length is present
        S bit is set to indicate Nr and Ns are present
        Version field is set to 0x02
        Length field is set to 0x14
        TunnelId is set to 0xD
        Sequence Number expected is set to 0x1
        Sequence Number is set to 0x1
        All other fields in the header are set to 0

    The following AVPs are sent:

    Message Type:
        Flags:  M is set to 0x1, length is set to 0x8, all other flags are 0x0
        VendorId is set to 0x0 (IETF)
        Attribute Type is set to 0x0 (Message type)
        Message Type in Attribute Value is set to 0x0003 (SCCCN)
        Length is set to 0x8

The client then sends ICRQ with the following details:

    Header:
        T bit is set to 1 to indicate control message
        L bit is set to 1 to indicate length is present
        S bit is set to indicate Nr and Ns are present
        Version field is set to 0x02
        Length field is set to 0x46
```

```
        TunnelId is set to 0xD
        Sequence Number expected is set to 0x1
        Sequence Number is set to 0x2
        All other fields in the header are set to 0

    The following AVPs are sent:

    Message Type:
        Flags:  M is set to 0x1, length is set to 0x8, all other flags are 0x0
        VendorId is set to 0x0 (IETF)
        Attribute Type is set to 0x0 (Message type)
        Message Type in Attribute Value is set to 0x000a (ICRQ)
        Length is set to 0x8
        Assigned Session Id:
        Flags:  M is set to 0x1, length is set to 0x8, all other flags are 0x0
        VendorId is set to 0x0 (IETF)
        Attribute Type is set to 0xE (Assigned Session Id)
        Assigned Session Id field is set to 0x01

    Call Serial Number:
        Flags:  M is set to 0x1, length is set to 0xa, all other flags are 0x0
        VendorId is set to 0x0 (IETF)
        Attribute Type is set to 0xF (Call Serial Number)
        Call Serial Number  is set to 0x0

    Bearer Type:
        Flags:  length is set to 0xa, all other flags are 0x0
        VendorId is set to 0x0 (IETF)
        Attribute Type is set to 0x12 (Bearer Type)
        Bearer Type: A bit is set to 0x1 (Analog channel), all other bits are 0x0

    Correlation Id:
        Flags: length is set to 0x2D, all other flags are 0x0
        VendorId is set to 0x137 (Microsoft)
        Attribute Type is set to 0x1 (Correlation Id)
        Attribute value in this example is the following 16 bit value:
          15 78 28 BF 3C 66 C0 4A 9D D9 6D 93 35 D4 32 B3 (randomly
          generated value)

The client logs the correlation ID so that tracing and debugging applications
can use this correlation ID to look at the logs on the server and correlate the
events on the client and server.

The server then responds with ICRP with the following
details:

    Header:
        T bit is set to 1 to indicate control message
        L bit is set to 1 to indicate length is present
        S bit is set to indicate Nr and Ns are present
        Version field is set to 0x02
        Length field is set to 0x28
        TunnelId is set to 0xD
        Session ID is set to 0x1
        Sequence Number expected is set to 0x3
        Sequence Number is set to 0x1
        All other fields in the header are set to 0

    The following AVPs are sent:
```

```
    Message Type:
        Flags:  M is set to 0x1, length is set to 0x8, all other flags are 0x0
        VendorId is set to 0x0 (IETF)
        Attribute Type is set to 0x0 (Message type)
        Message Type in Attribute Value is set to 0x000b (ICRP)
        Length is set to 0x8

    Assigned Session Id:
        Flags:  M is set to 0x1, length is set to 0x8, all other flags are 0x0
        VendorId is set to 0x0 (IETF)
        Attribute Type is set to 0xE (Assigned Session Id)
        Assigned Session Id field is set to 0x01

The server logs the correlation ID received in ICRP request in its trace logs
so that if the connection is torn down, an administrator will be able to find
out why a particular client connection could not go through.

The client then sends ICCN with the following details:

    Header:
        T bit is set to 1 to indicate control message
        L bit is set to 1 to indicate length is present
        S bit is set to indicate Nr and Ns are present
        Version field is set to 0x02
        Length field is set to 0x48
        TunnelId is set to 0xD
        Sequence Number expected is set to 0x2
        Sequence Number is set to 0x3
        All other fields in the header are set to 0

    The following AVPs are sent:

    Message Type:
        Flags:  M is set to 0x1, length is set to 0x8, all other flags are 0x0
        VendorId is set to 0x0 (IETF)
        Attribute Type is set to 0x0 (Message type)
        Message Type in Attribute Value is set to 0x000c (ICCN)
        Length is set to 0x8
        Tx Connection Speed:
        Flags:  M is set to 0x1, length is set to 0xa, all other flags are 0x0
        VendorId is set to 0x0 (IETF)
        Attribute Type is set to 0xE (Tx Connection Speed)
        Tx Connection Speed field is set to 0x5f5E100 (100 mbps)

    Framing type:
        Flags:  M is set to 0x1, length is set to 0xa, all other flags are 0x0
        VendorId is set to 0x0 (IETF)
        Attribute Type is set to 0x3 (Framing Type)
        S bit in Framing type is set to 0x1 rest are set to 0x0
        Proxy Authen Type:
        Flags:  length is set to 0xa, all other flags are 0x0
        VendorId is set to 0x0 (IETF)
        Attribute Type is set to 0x1D (Proxy Authen Type)
        Proxy Authen Type is set to 0x4  (No Authentication)

The client sends CDN in order to disconnect with the
following details:
```

```
    Header:
        T bit is set to 1 to indicate control message
        L bit is set to 1 to indicate length is present
        S bit is set to indicate Nr and Ns are present
        Version field is set to 0x02
        Length field is set to 0x38
        TunnelId is set to 0xD
        Sequence Number expected is set to 0x2
        Sequence Number is set to 0x4
        All other fields in the header are set to 0

    The following AVPs are sent:

    Message Type:
        Flags:  M is set to 0x1, length is set to 0x8, all other flags are 0x0
        VendorId is set to 0x0 (IETF)
        Attribute Type is set to 0x0 (Message type)
        Message Type in Attribute Value is set to 0x000E (CDN)
        Length is set to 0x8

    Assigned Session Id:
        Flags:  M is set to 0x1, length is set to 0x8, all other flags are 0x0
        VendorId is set to 0x0 (IETF)
        Attribute Type is set to 0xE (Assigned Session Id)
        Assigned Session Id field is set to 0x01

The client then sends StopCCN the following details:

    Header:
        T bit is set to 1 to indicate control message
        L bit is set to 1 to indicate length is present
        S bit is set to indicate Nr and Ns are present
        Version field is set to 0x02
        Length field is set to 0x26
        TunnelId is set to 0xD
        Sequence Number expected is set to 0x2
        Sequence Number is set to 0x5
        All other fields in the header are set to 0

    The following AVPs are sent:

    Message Type:
        Flags:  M is set to 0x1, length is set to 0x8, all other flags are 0x0
        VendorId is set to 0x0 (IETF)
        Attribute Type is set to 0x0 (Message type)
        Message Type in Attribute Value is set to 0x0004 (StopCCN)
        Length is set to 0x8
        Assigned Tunnel
        Flags:  M is set to 0x1, length is set to 0x8, all other flags are 0x0
        VendorId is set to 0x0 (IETF)
        Attribute Type is set to 0x9 (Assigned Tunnel)
        Assigned tunnelID is set to 0xd
        Length is set to 0x8
```

5 Security

5.1 Security Considerations for Implementers

All of the security considerations that are applicable to L2TP [RFC2661] and securing L2TP with IPsec [RFC3193] also apply to the Layer 2 Tunneling Protocol (L2TP) IPsec Extensions.

5.2 Index of Security Parameters

None.

6 Appendix A: Product Behavior

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

- Microsoft 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

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 2.1: Source port used is 0x6a5.

<2> Section 2.2.1: The use of Microsoft vendor-specific AVPs and the ATTR_VEN_MS_CorrID_Type are supported beginning with Windows Vista SP1 and Windows Server 2008.

<3> Section 3.1.3: The registry subkey used by Windows to store configuration data for WAN Miniportdevices using the Layer 2 Tunneling Protocol (L2TP) driver is HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Control\Class\{4D36E972-E325-11CE-BFC1-08002BE10318}\<Device-number>, where <Device-number> is the number of one of the 4-digit numbered subkeys in the {4D36E972-E325-11CE-BFC1-08002BE10318} subkey. The particular subkey differs between computers. To determine which of the numbered subkeys stores L2TP driver data on your computer, look for the value "Wan Miniport (L2TP)" (without quotes), in the DriverDesc entry in each numbered subkey.

<4> Section 3.1.5.1: The implementation sends the congestion control bit R as specified in section 4.3 of [L2TP draft]. The R bit sending mechanism can be controlled using the registry key HKLM\SYSTEM\CurrentControlSet\Control\Class\{4D36E972-E325-11CE-BFC1-08002BE10318}\<Device-number>\PayloadReceiveWindow.

If the key is absent, or is present with value 0, sequencing is disabled and hence the R bit is not sent.

<5> Section 3.1.5.1: The LAC/LNS accepts messages with sequence number Ns from Nr+1 up to Nr+Nx where Nx is a number giving how many out-of-order messages the receiver can buffer. The default value for Nx is 100 which can be changed by adding registry key HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Control\Class\{4D36E972-E325-11CE-BFC1-08002BE10318}\MaxOutOfOrder. The acceptable values for this registry key are between 0 and

0x4000 both inclusive. The newly received out-of-order message is queued for reordering. If it is a duplicate of an existing message in the out-of-order queue then it is silently discarded.

When a packet is received with Ns > Nr+Nx then it is silently dropped i.e. the state machine behaves as though it never received this packet.

<6> Section 3.1.5.3: The SCCRQ message sent by Windows LAC/LNS would have the following optional AV pairs:

Bearer Capabilities

Receive Window Size

Challenge

Firmware Revision

Vendor Name: The vendor name is "Microsoft" without quotes in ASCII encoding.

The SCCRQ message received at LAC/LNS does not check for the presence of the Host Name AV pair.

If the SCCRQ message received has a Bearer Capabilities AV pair, then the tunnel's Bearer Capability is set to the received value; else it is set to 0.

If the received SCCRQ message has a Receive Window AV pair, then the tunnel's Send Window is set to the received value, else the Send Window is set to 4.

The Tie Breaker AV pair is not processed by the LAC/LNS.

<7> Section 3.1.5.4: The following AV pairs are present in the SCCRP message sent by LAC/LNS as specified in section 6.2 of [RFC2661].

Bearer Capabilities

Firmware Revision

Vendor Name: The vendor name is "Microsoft" (without quotes) in ASCII encoding.

Receive Window Size: This is sent with the M bit set to 1. The value of the Receive Windows Size is configurable with the default being 4.

Challenge

Challenge Response

The SCCRP message received at LAC/LNS does not check for the presence of the Host Name AV pairs.

If the SCCRP message received has Bearer Capabilities AV pairs, then the tunnel's Bearer Capability is set to the received value; else it is set to 0.

If the received SCCRP message has Receive Window AV pairs, then the tunnel's Send Window is set to the received value; else the Send Window is set to 4.

<8> Section 3.1.5.5: The following AV pairs are present in the SCCCN message sent by an LAC/LNS:

Challenge Response: This AV pair is sent if a Challenge AV pair was received in the corresponding SCCRP message.

<9> Section 3.1.5.6: The LAC/LNS does not send an Error Message string in the Result Code AV pair of the StopCCN message.

<10> Section 3.1.5.7: This message is implemented by the LAC/LNS. The Hello Timer value (in milliseconds) can be set by using the registry key HKLM\SYSTEM\CurrentControlSet\Control\Class\{4D36E972-E325-11CE-BFC1-08002BE10318}\<Device-number>\HelloMs. If the registry key is present with value 0, then the Hello timer is disabled. If the registry key is absent, then the default value used is 40 seconds.

<11> Section 3.1.5.8: The Result Code AV pair sent by the LAC/LNS does not contain an Error Message string.

<12> Section 3.2.1: The LAC/LNS neither sends the Tie Breaker AVP nor processes it. If the TunnelId present in the SCCRQ message received from the peer is the same as the TunnelId sent to the peer, the SCCRQ message is ignored and the state machine does not transition from the wait-ctl-reply state. If the TunnelId present in the SCCRQ message received from the peer is different from the TunnelId sent to the peer, the SCCRQ message received is treated as a new control connection request. The state machine for the new control connection will transition from idle to wait-ctl-conn state. This will result in two control channels which are agnostic of each other. Call sessions can be negotiated over either of these control tunnels. Since each tunnel provides a data channel between the LAC/LNS, the data packets can flow over either of them. The choice of the data channel for sending or receiving data packets is not deterministic.

<13> Section 3.2.5.1: The LAC sends the following AV pairs in the ICRQ message:

Bearer Type

Physical Channel ID

The LNS server ignores the Bearer Type AVP, and the server's behavior is unaffected by the presence or absence of a Bearer Type AVP.

If the LNS does not receive a Physical Channel ID AVP, it assumes a value of 0xFFFFFFFF.

<14> Section 3.2.5.2: The LAC sends the following AV pairs in the ICCN message:

Proxy Authen Type (Proxy Authen Type = No Authentication = 4)

Sequencing Required

Receive Window Size

<15> Section 3.2.5.3: LAC sends the Physical Channel ID AV pair in the OCRP message. If the LNS does not receive a Physical Channel ID it assumes a value of 0xFFFFFFFF.

<16> Section 3.2.5.4: The LAC sends a Receive Window Size AV pair in the OCCN message. The AV pair is sent with the M bit set to 1. The LAC sends the Sequencing Required AV pair in the OCCN message. The behavior of the LNS is defined in section 5.3 of [L2TP draft].

<17> Section 3.2.5.5: The LAC does not send this message.

<18> Section 3.3.5.1: The LNS sends the Receive Window Size AV pair in the ICRP message if the registry key HKLM\SYSTEM\CurrentControlSet\Control\Class\{4D36E972-E325-11CE-BFC1-08002BE10318}\<Device-number>\PayloadReceiveWindowhas a nonzero value. If the registry key is absent or has value zero, then the LNS does not send the Receive Window Size AV pair in the

ICRP message. The LAC, not the LNS, checks for the absence of any of the required AV pairs in the ICRP message returned in response to the ICRQ (see section 6.7 of [RFC2661]).

<19> Section 3.3.5.2: The LNS does not check for the absence of any of the required AV pairs in the ICCN message. If the (Tx) Connect Speed AV pair is absent in the received ICCN message, then the Windows LNS uses 9600 bps as the default value.

The LNS supports only Synchronous Framing. The LNS supports only Synchronous Framing. The behavior when any other value is received in the Framing Type AV pair in ICCN message depends on the value set for the registry key HKLM\SYSTEM\CurrentControlSet\Control\Class\{4D36E972-E325-11CE-BFC1-08002BE10318}\<Device-number>\IgnoreFramingMismatch, as defined in the following table:

Value	Meaning
Key not present	The error is ignored.
0	Do not ignore framing mismatch. The tunnel is terminated.
Any nonzero value	The error is ignored.

If the Receive Window Size AV pair is not received in the ICCN message, then the LNS does not support sequencing on this call. If the Receive Window Size AV pair is received with value 0, then the LNS uses a Send Window of 10000.

The LNS accepts the Packet Processing Delay AV pair [L2TP draft] (not as defined in IETF RFC2661) and derives Round Trip milliseconds from the value of this AV pair.

<20> Section 3.3.5.3: The Windows LNS sends the Receive Window Size AV pair with M bit set to 1 in the OCRQ message if it is configured to do so.

If the registry key HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Control\Class\{4D36E972-E325-11CE-BFC1-08002BE10318}\<Device-number>\PayloadReceiveWindow is present and has a non-zero value, then a Receive Window Size AV pair is sent by the Windows LNS. If this registry key is absent, which is the default case, or if the key has value zero, then the Receive Window Size AVP is not sent.

<21> Section 3.3.5.4: The Windows LNS disconnects the tunnel by sending a StopCCN message when it receives a zero value for Assigned Session ID AV pair in OCRP message.

<22> Section 3.3.5.5: Windows LNS does not check for the absence of any of the required AV pairs of the OCCN message.

<23> Section 3.3.5.6: Windows LNS does not check for the absence of any of the required AV pairs of the WEN message.

7 Change Tracking

This section identifies changes that were made to the [MS-L2TPIE] protocol document between the May 2011 and June 2011 releases. Changes are classified as New, Major, Minor, Editorial, or No change.

The revision class **New** means that a new document is being released.

The revision class **Major** means that the technical content in the document was significantly revised. Major changes affect protocol interoperability or implementation. Examples of major changes are:

- A document revision that incorporates changes to interoperability requirements or functionality.
- An extensive rewrite, addition, or deletion of major portions of content.
- The removal of a document from the documentation set.
- Changes made for template compliance.

The revision class **Minor** means that the meaning of the technical content was clarified. Minor changes do not affect protocol interoperability or implementation. Examples of minor changes are updates to clarify ambiguity at the sentence, paragraph, or table level.

The revision class **Editorial** means that the language and formatting in the technical content was changed. Editorial changes apply to grammatical, formatting, and style issues.

The revision class **No change** means that no new technical or language changes were introduced. The technical content of the document is identical to the last released version, but minor editorial and formatting changes, as well as updates to the header and footer information, and to the revision summary, may have been made.

Major and minor changes can be described further using the following change types:

- New content added.
- Content updated.
- Content removed.
- New product behavior note added.
- Product behavior note updated.
- Product behavior note removed.
- New protocol syntax added.
- Protocol syntax updated.
- Protocol syntax removed.
- New content added due to protocol revision.
- Content updated due to protocol revision.
- Content removed due to protocol revision.
- New protocol syntax added due to protocol revision.

- Protocol syntax updated due to protocol revision.
- Protocol syntax removed due to protocol revision.
- New content added for template compliance.
- Content updated for template compliance.
- Content removed for template compliance.
- Obsolete document removed.

Editorial changes are always classified with the change type **Editorially updated.**

Some important terms used in the change type descriptions are defined as follows:

- **Protocol syntax** refers to data elements (such as packets, structures, enumerations, and methods) as well as interfaces.
- **Protocol revision** refers to changes made to a protocol that affect the bits that are sent over the wire.

The changes made to this document are listed in the following table. For more information, please contact protocol@microsoft.com.

Section	Tracking number (if applicable) and description	Major change (Y or N)	Change type
1.2 References	Added explanatory statement regarding the removal of the publishing year from Microsoft Open Specification document references.	N	Content updated.

8 Index

T

V