[MS-GRVWDPP]:
Wide Area Network Device Presence Protocol (WAN DPP) Specification

---

Revision Summary

Date	Revision History	Revision Class	Comments
04/04/2008	0.1		Initial Availability
06/27/2008	1.0	Major	Revised and edited the technical content
10/06/2008	1.01	Editorial	Revised and edited the technical content
12/12/2008	1.02	Editorial	Revised and edited the technical content
07/13/2009	1.03	Major	Revised and edited the technical content
08/28/2009	1.04	Editorial	Revised and edited the technical content
11/06/2009	1.05	Editorial	Revised and edited the technical content
02/19/2010	2.0	Minor	Updated the technical content
03/31/2010	2.01	Editorial	Revised and edited the technical content
04/30/2010	2.02	Editorial	Revised and edited the technical content
06/07/2010	2.03	Editorial	Revised and edited the technical content
06/29/2010	2.04	Editorial	Changed language and formatting in the technical content.
07/23/2010	2.05	Minor	Clarified the meaning of the technical content.
09/27/2010	2.05	No change	No changes to the meaning, language, or formatting of the technical content.
11/15/2010	2.05	No change	No changes to the meaning, language, or formatting of the technical content.
12/17/2010	2.05	No change	No changes to the meaning, language, or formatting of the technical content.
03/18/2011	2.6	Minor	Clarified the meaning of the technical content.
06/10/2011	2.6	No change	No changes to the meaning, language, or formatting of the technical content.
01/20/2012	3.0	Major	Significantly changed the technical content.

Table of Contents

1 Introduction

This document specifies the Wide Area Network Device Presence Protocol (WAN DPP). This protocol is used by clients and servers to enable clients to discover and acquire the presence information of other cooperating devices. WAN DPP is a message-based protocol that requires a network transport that guarantees reliable and ordered delivery.

The WAN DPP protocol supports the following capabilities:

- Client publication of its presence information to a server.
- Subscription to presence information.
- Server notification of published presence information to subscribing clients.

This document specifies the following:

- How messages are encapsulated on the wire, common data types, and requirements in section 2.
- Protocol details, including client and server details, in section 3.
- Protocol examples in section 4.
- Security considerations for implementers in section 5.
- Product Behavior in appendix A.

Sections 1.8, 2, and 3 of this specification are normative and contain RFC 2119 language. Sections 1.5 and 1.9 are also normative but cannot contain RFC 2119 language. All other sections and examples in this specification are informative.

1.1 Glossary

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

ASCII
big-endian
Internet Protocol version 4 (IPv4)
Internet Protocol version 6 (IPv6)
little-endian
network address translation (NAT)
Transmission Control Protocol (TCP)

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

connection
device
device URL
empty string
presence
presence information
presence server
publish
session
Simple Symmetric Transport Protocol (SSTP)

The following terms are specific to this document:

notify: The process of sharing presence information with subscribed client devices by using the Wide Area Network Device Presence Protocol (WAN DPP).

subscribe: The process of registering to receive updates about presence information for client devices. The updates are delivered by using Wide Area Network Device Presence Protocol (WAN DPP).

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-GRVSSTP] Microsoft Corporation, "Simple Symmetric Transport Protocol (SSTP) Specification".

[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

1.2.2 Informative References

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

[MS-GRVHENC] Microsoft Corporation, "HTTP Encapsulation of Simple Symmetric Transport Protocol (SSTP) Protocol Specification".

[MS-OFCGLOS] Microsoft Corporation, "Microsoft Office Master Glossary".

1.3 Protocol Overview (Synopsis)

The Wide Area Network Device Presence Protocol (WAN DPP) supports the publication and discovery of **presence information** for client **devices**. WAN DPP uses **Simple Symmetric Transport Protocol (SSTP)** as its transport, as described in [MS-GRVSSTP]. This specification defines WAN DPP versions 4.1 and 5.0.

WAN DPP uses a publish-subscribe model where clients **publish (2)** their device information to WAN DPP servers and **subscribe** to WAN DPP servers for the published device information of specified devices. WAN DPP servers **notify** subscribed clients of updates to published **presence (1)** information.

Device URLs identify WAN DPP client devices. Each WAN DPP client is assigned to a **presence server**. The mechanism for determining device URLs and presence servers is application-dependent and is outside the scope of this document.

Device presence information consists of a client device's online status, the TCP port number and IP addresses on which the client device listens for SSTP messages, and other information that can be used by peer clients to determine connectivity.

The WAN DPP system involves the following roles:

- A publishing client that sends its device presence information to a presence server.
- A subscribing client that subscribes to device presence information stored on a presence server.
- A presence server that stores publishing clients' presence information and notifies subscribing clients of updates to this information.

Any WAN DPP device can implement any combination of these roles.

These roles are associated with WAN DPP messages, as described in the following sections.

1.3.1 Messages

WAN DPP messages are as follows:

- Publish – This message is sent from a client to a server to post device presence information.
- Subscribe – This message is sent to a server to request device presence information for specific devices that publish (2) their online status on the server.
- Unsubscribe – This message is sent to a server, instructing the server to stop sending Notify messages for a device to which the client previously subscribed.
- Notify – This message is sent from a presence server in response to a subscription message, providing presence information for the requested devices. Whenever the presence information for a device changes, the server sends Notify messages.
- Noop– This message is sent from a client to a server. It has no purpose relevant to WAN DPP.
- VersionRejected – This message is sent from a server to a client to support protocol version negotiation.

1.3.2 Example Message Flow

The following diagram illustrates a basic WAN DPP message flow between two client devices and a presence server, where Client 2 sends a Subscription to the presence server requesting the published device presence information of Client 1.

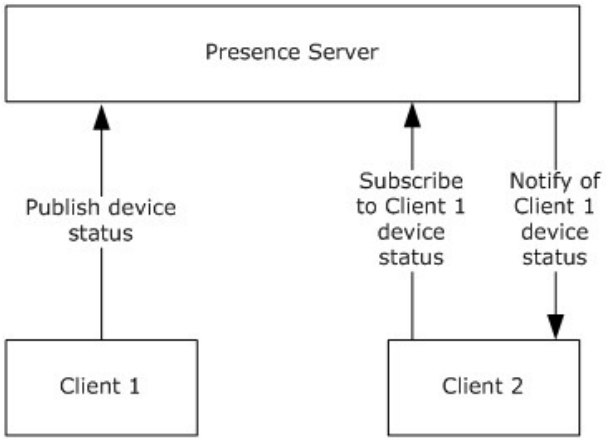


Figure 1: WAN DPP Message Flow

1.4 Relationship to Other Protocols

WAN DPP depends upon the underlying application-level SSTP protocol on client and server devices. The SSTP protocol guarantees message delivery and preserves the order of WAN DPP messages.

WAN DPP uses Simple Symmetric Transport Protocol (SSTP) as its transport, as described in [MS-GRVSSTP]. The **session (1)** capabilities of SSTP enable WAN DPP to establish bi-directional communications between a client and a presence server. All messages sent from the client to the presence server flow over one SSTP session; messages from the presence server to the client flow over another session. The SSTP application layer transport can be encapsulated in a standard HTTP connection over port 80/TCP, as described in [MS-GRVHENC].

The following diagram shows the underlying messaging and transport stack used by the protocol:

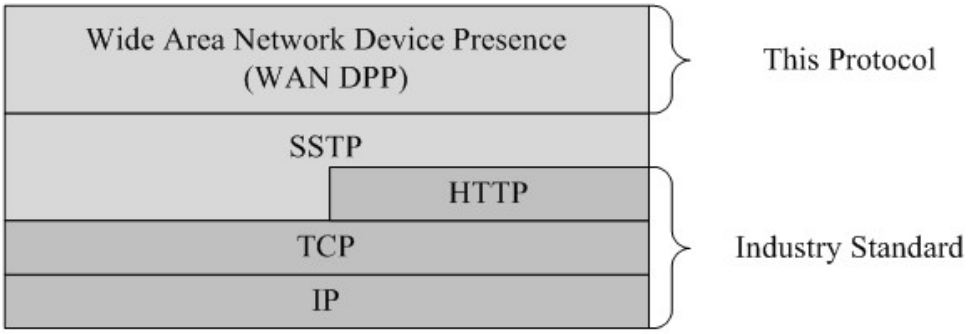


Figure 2: This protocol in relation to other protocols

1.5 Prerequisites/Preconditions

WAN DPP requires client devices to be assigned a presence server and a Device URL by a higher-level application or protocol.

1.6 Applicability Statement

Presence servers collect and store the device addresses and online status of published clients. Applications can use WAN DPP to exchange presence information between devices.

1.7 Versioning and Capability Negotiation

This document covers versioning issues in the following areas:

- Supported Transports: WAN DPP uses SSTP as its transport.
- Protocol Versions: For an established SSTP **connection (1)** which is using SSTP version 1.5, as determined by the SSTP active connection Version state variable described in [MS-GRVSSTP], WAN DPP Version 4.1 is used. For an SSTP connection using SSTP 1.6, WAN DPP Version 5.0 is used.
- Security and Authentication Methods: WAN DPP relies on SSTP for security and authentication, as described in [MS-GRVSSTP].
- Capability Negotiation: WAN DPP supports limited version negotiation using the VersionRejected message described in section 3.

1.8 Vendor-Extensible Fields

None.

1.9 Standards Assignments

None.

2 Messages

2.1 Transport

WAN DPP messages are sent over SSTP sessions between two devices. SSTP sessions depend on an established SSTP connection (1). Because each SSTP session carries messages in only one direction, two sessions are required for bi-directional exchange. Creating an SSTP session requires three parameters: ResourceURL, IdentityURL, and DeviceURL. WAN DPP sessions are created using specific values for these parameters. For more details about opening SSTP connections and sessions, see [MS-GRVSSTP].

For any session opened to a presence server<1>:

- The ResourceURL MUST be "grooveWanDPP".
- The IdentityURL MUST be an **empty string (1)**.
- The DeviceURL MUST be the URL that identifies the connecting client.

For a session that opened by a presence server to a client:

- The ResourceURL MUST be "grooveWanDPP".
- The IdentityURL MUST be an empty string (1).
- The DeviceURL MUST be an empty string (1).

Each WAN DPP message is sent within one or more SSTP Data commands, depending on the size of the WAN DPP message. The SSTP Data maximum payload size is 2048 bytes. WAN DPP messages exceeding this size are sent in multiple SSTP Data commands<2>.

2.2 Message Syntax

This section specifies the six message types that comprise the WAN DPP protocol: Publish, Subscribe, Unsubscribe, Notify, VersionRejected, and Noop.

WAN DPP messages share three common parameters, indicated in the first three bytes of every message: major version, minor version, and message type, in that order. All three fields are unsigned integers.

The first two common parameters indicate the WAN DPP protocol version. These fields MUST indicate either version 4.1 or version 5.0, the two versions that are specified in this document. In WAN DPP 4.1 messages the first byte of each WAN DPP message MUST be 0x04 and the second byte MUST be 0x01. In WAN DPP 5.0 messages the first byte of each WAN DPP message MUST be 0x05 and the second byte MUST be 0x00. The third common parameter is the Message Type field, which indicates the type of WAN DPP message contained by the SSTP message. The following table lists the six valid WAN DPP message types:

Value	Message type
0x00	Publish
0x01	Subscribe
0x02	Unsubscribe

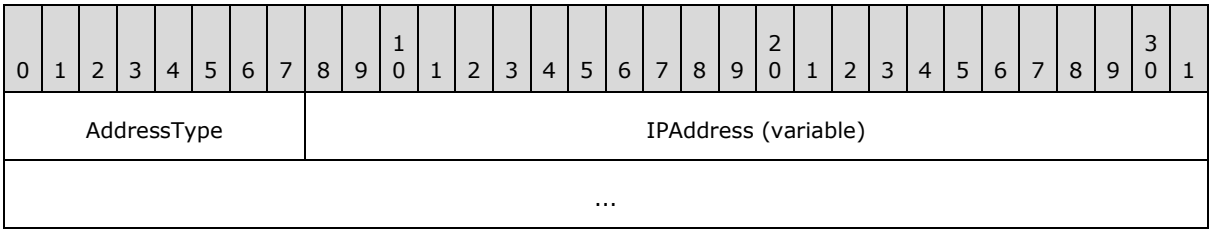
Value	Message type
0x03	Notify
0x04	Noop
0x06	VersionRejected

Publish and Notify messages both contain an IPAddresses field that contains a list of IP addresses. The number of elements in this list MUST be equal to the value in the NumberOfIPAddr field. If NumberOfIPAddr is zero then IPAddresses MUST be 0x00 indicating an empty list. If NumberOfIPAddr is not equal to zero, IPAddresses MUST contain a list of IP Addresses encoded according to the following rules:

The WAN DPP 4.1 version of IPAddresses only supports **IPv4** addresses that are encoded in **little-endian** long format. For example, an address of "1.2.3.4" would be encoded as a stream of 4 consecutive bytes: 0x04 0x03 0x02 0x01.

The WAN DPP 5.0 version supports **IPv6** addresses, and they are encoded in **big-endian** long format. For example, an IPv6 address of "102:304:506:708:90A:B0C:D0E:F10" would be encoded as a stream of 16 consecutive bytes: 0x01 0x02 0x03 0x04 0x05 0x06 0x07 0x08 0x09 0x0A 0x0B 0x0C 0x0D 0x0E 0x0F 0x10.

In WAN DPP 5.0 messages that contain IP addresses, the addresses are encoded in an IPAddressesV5 field. IPAddressesV5 are defined as shown in the following table.



AddressType (1 byte): The type of IP address (either IPv4 or IPv6). This field MUST be either 0x01 or 0x02. If the address specified by the IPAddress field is an IPv4 address this field MUST be 0x01, if that address is an IPv6 address this field MUST be 0x02.

IPAddress (variable): This field MUST be either an IPv4 address encoded in little-endian long format (4 bytes), or an IPv6 address encoded in big-endian long format (16 bytes).

Subscribe, **Unsubscribe**, and **Notify** messages can contain information for multiple devices. To support specification of multiple devices, the protocol contains one field with a count of items, followed by a set of fields that repeats the specified number of times.

All fixed-length fields are interpreted as unsigned integers and have their sizes specified later in this section. All multi-byte fields use little endian byte order. Multi-byte integer fields are not required to align on 2 or 4-byte boundaries but MUST align on byte boundaries.

The maximum length of a WAN DPP message is 4096 bytes.

2.2.1 Publish for WAN DPP 4.1

The Publish message informs the presence server of a client device's presence information. A publishing client sends this message to a presence server when the client first starts up and whenever the client presence information changes – for example, if IP addresses change. The

Publish message fields, with the exception of MajorVersion, MinorVersion, and MessageType, supply the presence information for the publishing client.

Publish fields are defined as shown in the following table.

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
MajorVersion								MinorVersion								MessageType								Status							
NumberOfIPAddr								IPAddresses (variable)																							
...																															
ClientSSTPPort																DPPSessionID															
...																ClientPlatformVersion (variable)															
...																															

MajorVersion (1 byte): The WAN DPP major version. This value MUST be 0x04.

MinorVersion (1 byte): The WAN DPP minor version. This value MUST be 0x01.

MessageType (1 byte): The WAN DPP message type. For a Publish message, this field MUST be 0x00.

Status (1 byte): Current online or offline status of the device that is sending the Publish message. The value MUST be either 0x80 to publish status as online, or 0x00 to publish status as offline.

NumberOfIPAddr (1 byte): The number of IP addresses for which this message is publishing presence information.

IPAddresses (variable): A list of IPv4 addresses on which the publishing client device listens for SSTP protocol messages. The NumberOfIPAddr field specifies the number of times that this 4-byte field repeats.

ClientSSTPPort (2 bytes): The **TCP** port number on which the client device listens for SSTP protocol messages.

DPPSessionID (4 bytes): Identifies an instance of a device's online status.

ClientPlatformVersion (variable): The version of the application running on the client device. This value MUST be an **ASCII** string terminated by 0x00.

2.2.2 Publish for WAN DPP 5.0

The Publish message informs the presence server of a client device's presence information. A publishing client sends this message to a presence server when the client first starts up and whenever the client presence information changes – for example, if IP addresses change. The Publish message fields, with the exception of MajorVersion, MinorVersion, and MessageType, supply the presence information for the publishing client. This message supports both IPv4 and IPv6 addresses.

Publish fields are defined as shown in the following table.

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
MajorVersion								MinorVersion								MessageType								Status							
NumberOfIPAddr								IPAddressesV5 (variable)																							
...																															
ClientSSTPPort																DPPSessionID															
...																ClientPlatformVersion (variable)															
...																															

MajorVersion (1 byte): The WAN DPP major version. This value MUST be 0x05.

MinorVersion (1 byte): The WAN DPP minor version. This value MUST be 0x00.

MessageType (1 byte): The WAN DPP message type. For a **Publish** message, this field MUST be 0x00.

Status (1 byte): Current online or offline status of the device that is sending the **Publish** message. The value MUST be either 0x80 to publish status as online, or 0x00 to publish status as offline.

NumberOfIPAddr (1 byte): The number of IP addresses for which this message is publishing presence information.

IPAddressesV5 (variable): A list of IP addresses (IPv4 or IPv6) on which the publishing client device listens for SSTP protocol messages. The NumberOfIPAddr field specifies the number of times that this field repeats. For more information about how the IP addresses are encoded in this list see section 2.2.

ClientSSTPPort (2 bytes): The TCP port number on which the client device listens for SSTP protocol messages.

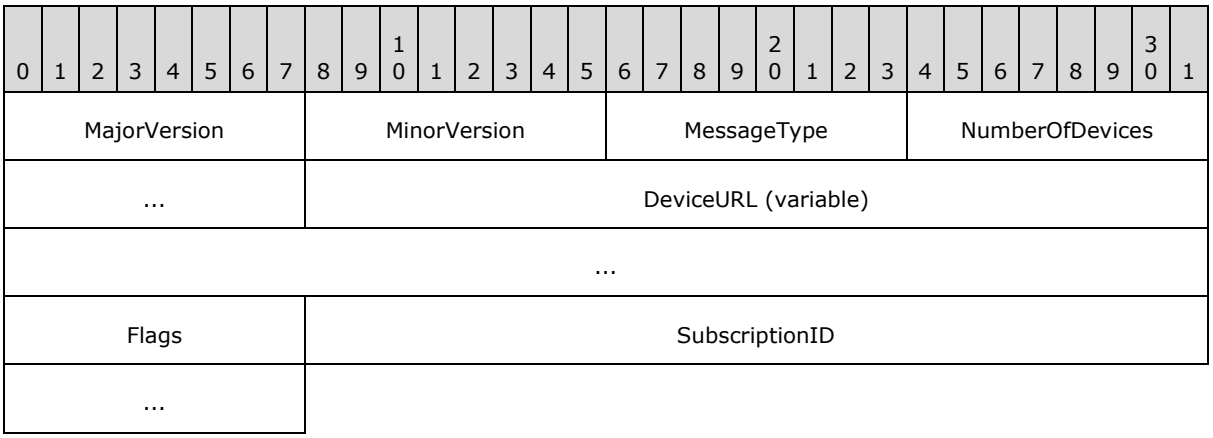
DPPSessionID (4 bytes): Identifies an instance of a device's online status.

ClientPlatformVersion (variable): The version of the application running on the client device. This value MUST be an ASCII string terminated by 0x00.

2.2.3 Subscribe for WAN DPP 4.1

The **Subscribe** message requests notification of updates to the presence information of specified client devices. A subscribing client sends this message to a presence server.

Subscribe fields are defined as shown in the following table.



MajorVersion (1 byte): The WAN DPP major version. This field MUST be 0x04.

MinorVersion (1 byte): The WAN DPP minor version. This field MUST be 0x01.

MessageType (1 byte): The WAN DPP message type. For a Subscribe message, this field MUST be 0x01.

NumberOfDevices (2 bytes): The number of devices whose presence information is requested. The value in this field specifies the number of times that the following three fields (DeviceURL, Flags, and SubscriptionID) are repeated in the contents of this message. When this value is greater than 1 the fields are repeated in order (DeviceURL1, Flags1, SubscriptionID1, DeviceURL2, Flags2, SubscriptionID2… DeviceURLn, Flagsn, SubscriptionIDn). The value in this field is limited by the 4096-byte maximum message size for all WAN DPP messages.

DeviceURL (variable): The **device URL** for the device whose presence information is requested. This field is an ASCII string terminated by 0x00.

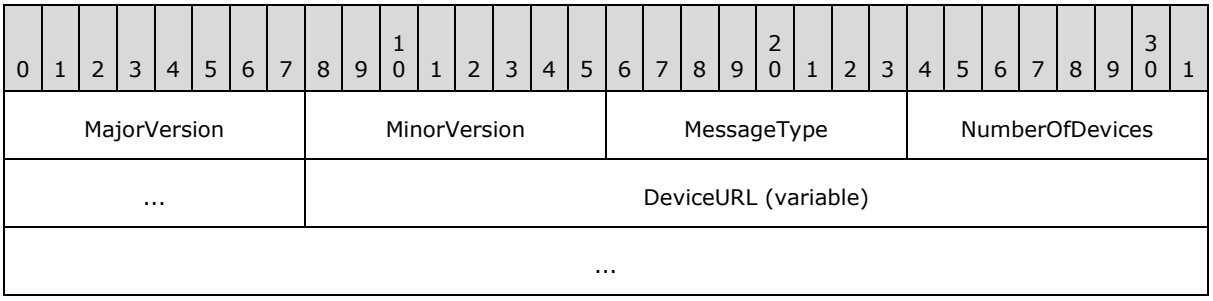
Flags (1 byte): This field is reserved. This field MUST contain a value of 0x00.

SubscriptionID (4 bytes): An identifier generated on the subscribing client. The presence server associates this **SubscriptionID** with the **DeviceURL** and returns this identifier in **Notify** messages for that device.

2.2.4 Subscribe for WAN DPP 5.0

The **Subscribe** message requests notification of updates to the presence information of specified devices. This message is sent to a presence server.

Subscribe fields are defined as shown in the following table.



EndServerURL (variable)	
...	
Flags	SubscriptionID
...	

MajorVersion (1 byte): The WAN DPP major version. This field MUST be 0x05.

MinorVersion (1 byte): The WAN DPP minor version. This field MUST be 0x00.

MessageType (1 byte): The WAN DPP message type. For a **Subscribe** message, this field MUST be 0x01.

NumberOfDevices (2 bytes): The number of devices whose presence information is requested. The value in this field specifies the number of times that the following four fields (**DeviceURL**, **EndServerURL**, **Flags**, and **SubscriptionID**) are repeated in the contents of this message. When this value is greater than 1 the fields are repeated in order (DeviceURL1, EndServerURL1, Flags1, SubscriptionID1, DeviceURL2, EndServerURL2, Flags2, SubscriptionID2… DeviceURLn, EndServerURLn, Flagsn, SubscriptionIDn). The value in this field is limited by the 4096-byte maximum message size for all WAN DPP messages.

DeviceURL (variable): The device URL for the device whose presence information is requested. This field is an ASCII string terminated by 0x00.

EndServerURL (variable): This field is reserved. It SHOULD contain a value of 0x00. Any value in this field MUST be terminated by 0x00.

Flags (1 byte): This field is reserved. The Flags field MUST contain a value of 0x00.

SubscriptionID (4 bytes): An identifier generated on the subscribing client. The presence server associates this **SubscriptionID** with the **DeviceURL** and returns this identifier in **Notify** messages for that device.

2.2.5 Unsubscribe for WAN DPP 4.1

The **Unsubscribe** message informs the presence server that the client no longer requests notification of presence (1) updates. A subscribing client sends this message to a presence server. This message specifies a list of devices to whose presence information the subscribing client had previously subscribed.

Unsubscribe fields are defined as shown in the following table.

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
MajorVersion								MinorVersion								MessageType								NumberOfDevices							
...								DeviceURL (variable)																							
...																															

Flags	SubscriptionID
...	

MajorVersion (1 byte): The WAN DPP major version. This value MUST be 0x04.

MinorVersion (1 byte): The WAN DPP minor version. This value MUST be 0x01.

MessageType (1 byte): The WAN DPP message type. For an Unsubscribe message, this field MUST be 0x02.

NumberOfDevices (2 bytes): The number of devices whose presence information is no longer requested. The value in this field specifies the number of times that the following three fields (**DeviceURL**, **Flags**, and **SubscriptionID**) are repeated in the contents of this message. When this value is greater than 1 the fields are repeated in order (DeviceURL1, Flags1, SubscriptionID1, DeviceURL2, Flags2, SubscriptionID2… DeviceURLn, Flagsn, SubscriptionIDn). The value in this field is limited by the 4096-byte maximum length for all WAN DPP messages.

DeviceURL (variable): The device URL for the device to whose presence information the sender is unsubscribing. This field is an ASCII string terminated by 0x00.

Flags (1 byte): A reserved field. This field MUST contain a value of 0x00.

SubscriptionID (4 bytes): An identifier that was generated on the subscribing client when it sent the subscribe message for this subscription. It can also be zero, which means that the presence server MUST remove all entries associated with the **DeviceURL**, regardless of this field.

2.2.6 Unsubscribe for WAN DPP 5.0

The **Unsubscribe** message informs the presence server that the client no longer requests notification of presence updates. This message is sent to a presence server. The subscriptions for which updates are no longer requested are identified by a list of SubscriptionIDs.

Unsubscribe fields are defined as shown in the following table.

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
MajorVersion								MinorVersion								MessageType								NumberOfDevices							
...								DeviceURL (variable)																							
...																															
EndServerURL (variable)																															
...																															
Flags								SubscriptionID																							

...

MajorVersion (1 byte): The WAN DPP major version. This value MUST be 0x05.

MinorVersion (1 byte): The WAN DPP minor version. This value MUST be 0x00.

MessageType (1 byte): The WAN DPP message type. For an **Unsubscribe** message, this field MUST be 0x02.

NumberOfDevices (2 bytes): The number of devices whose presence information is no longer requested. The value in this field specifies the number of times that the following four fields (**DeviceURL**, **EndServerURL**, **Flags**, and **SubscriptionID**) are repeated in the contents of this message. When this value is greater than 1 the fields are repeated in order (DeviceURL1, EndServerURL1, Flags1, SubscriptionID1, DeviceURL2, EndServerURL2, Flags2, SubscriptionID2… DeviceURLn, EndServerURLn, Flagsn, SubscriptionIDn). The value in this field is limited by the 4096-byte maximum length for all WAN DPP messages.

DeviceURL (variable): This field value MUST be 0x00.

EndServerURL (variable): This field value MUST be 0x00.

Flags (1 byte): A reserved field. This field MUST contain a value of 0x00.

SubscriptionID (4 bytes): An identifier that was generated on the subscribing client when it sent the subscribe message for this subscription.

2.2.7 Notify for WAN DPP 4.1

The **Notify** message provides subscribing clients with the latest presence information for specified devices. A presence server sends this message to a subscribing client, to announce an update to the online status of a published device.

Notify fields are defined as shown in the following table.

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
MajorVersion								MinorVersion								MessageType								NumberOfDevices							
...								DeviceURL (variable)																							
...																															
SubscriptionID																															
Status								NumberOfIPAddr								IPAddresses (variable)															
...																															
ClientSSTPPort																TranslatedIP															

...	TranslatedPort
DPPSessionID	
ClientPlatformVersion (variable)	
...	

MajorVersion (1 byte): Indicates the WAN DPP major version. This value MUST be 0x04.

MinorVersion (1 byte): Indicates the WAN DPP minor version. This value MUST be 0x01.

MessageType (1 byte): Indicates a WAN DPP message type. For a **Notify** message, this field MUST be 0x03.

NumberOfNotifications (2 bytes): The number of devices whose presence information is included in this message. This field indicates the number of times that the remaining fields are repeated in the contents of this message. When this value is greater than 1 the fields are repeated in order (DeviceURL1, SubscriptionID1, …, DeviceURL2, SubscriptionID2…, … DeviceURLn, SubscriptionIDn, …n).The value of this field is limited by the 4096-byte message size limit for all WAN DPP messages.

DeviceURL (variable): The device URL for the device whose presence information is being updated with this message. This field is an ASCII string terminated by 0x00.

SubscriptionID (4 bytes): The **SubscriptionID** that was sent by the subscribing client in the **Subscribe** message for the device identified by the **DeviceURL**.

The remaining fields contain the presence information associated with the **DeviceURL**. This information is supplied by publishing clients or is set by the server itself when a publishing client goes offline.

Status (1 byte): The current online/offline status of the device. The value of this field MUST be either 0x80 if the device is online, or 0x00 if the device is offline.

NumberOfIPAddr (1 byte): An integer that represents the number of IP addresses for the device.

IPAddresses (variable): A list of IPv4 addresses for the device. The NumberOfIPAddr field specifies the number of times this 4-byte field repeats.

ClientSSTPPort (2 bytes): The TCP port on which the client device listens for SSTP messages.

TranslatedIP (4 bytes): The device's IP address as recognized by the presence server that is sending this **Notify** message.

For example: if the device is behind a **network address translation (NAT)** device, the IP addresses specified in the IPAddresses field are those which the device has been assigned, behind the NAT. But the TranslatedIP is the IP address of the NAT device.

TranslatedPort (2 bytes): The outbound TCP port for the device's SSTP connection to the server, as recognized by the presence server that is sending this notify message.

For example, if the device is behind a NAT device, the TCP port specified in the ClientSSTPPort field is the port on which the device listens, but the **TranslatedPort** is the outbound TCP port of the NAT device.

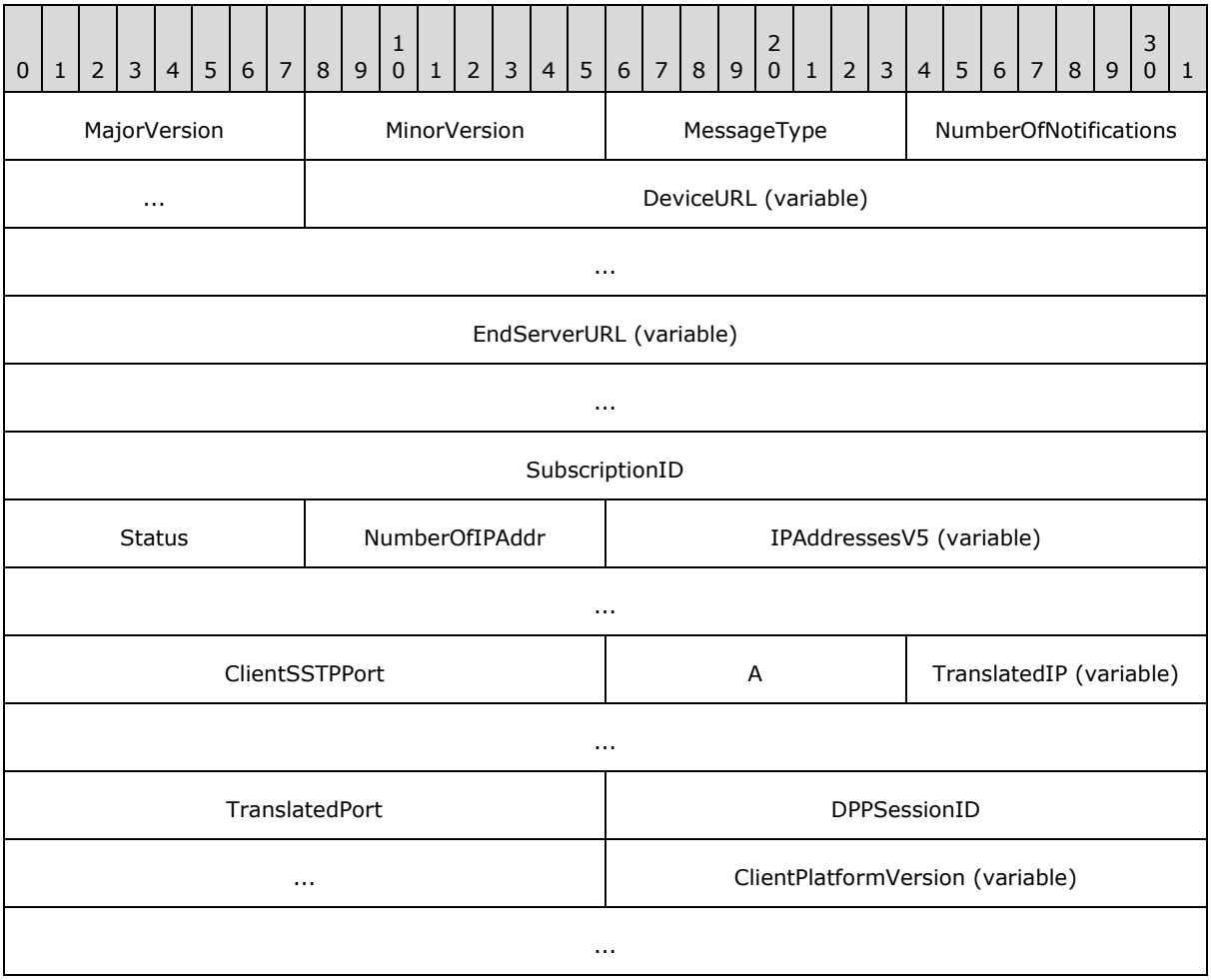
DPPSessionID (4 bytes): The identifier that the publishing client provided with the **Publish** message.

ClientPlatformVersion (variable): An ASCII string terminated by 0x00 that the publishing client provided with the **Publish** message.

2.2.8 Notify for WAN DPP 5.0

The **Notify** message provides subscribing clients with the latest presence information for specified devices. A presence server sends this message to announce an update to the online status of a device.

Notify fields are defined as shown in the following table.



MajorVersion (1 byte): Indicates the WAN DPP major version. This value MUST be 0x05.

MinorVersion (1 byte): Indicates the WAN DPP minor version. This value MUST be 0x00.

MessageType (1 byte): Indicates a WAN DPP message type. For a **Notify** message, this field MUST be 0x03.

NumberOfNotifications (2 bytes): The number of devices whose presence information is included in this message. This field indicates the number of times that the remaining fields are repeated in the contents of this message. When this value is greater than 1 the fields are repeated in order (DeviceURL1, EndServerURL1, …, DeviceURL2, EndServerURL2…, … DeviceURLn, EndServerURLn, …n). The value of this field is limited by the 4096-byte message size limit for all WAN DPP messages.

DeviceURL (variable): This field value MUST be 0x00.

EndServerURL (variable): This field value MUST be 0x00.

SubscriptionID (4 bytes): The **SubscriptionID** in this message MUST equal a **SubscriptionID** that the subscribing client sent to the presence server in an earlier **Subscribe** message.

The remaining fields contain the presence information associated with the device that the Subscribing client identified in the Subscribe message that contained this SubscriptionID. This information is supplied by publishing clients or is set by the server itself when a device goes offline or becomes available.

Status (1 byte): The current online/offline status of the device. The value of this field MUST be either 0x80 if the device is online, or 0x00 if the device is offline.

NumberOfIPAddr (1 byte): An integer that represents the number of IP addresses for the device.

IPAddressesV5 (variable): A list of IP addresses (IPv4 or IPv6) for the device. The NumberOfIPAddr field specifies the number of times that this field repeats. For more information about how the IP addresses are encoded in this list see section 2.2.

ClientSSTPPort (2 bytes): The TCP port on which the client device listens for SSTP messages.

A - NumberOfTranslatedIPAddr (1 byte): This field MUST contain a value of 0x01.

TranslatedIP (variable): The device's IP address as recognized by the presence server that is sending this Notify message. For more information about how this IPv4 (5 bytes) or IPv6 (17 bytes) address is encoded see section 2.2.

For example: if the device is behind a network address translation (NAT) device, the IP addresses specified in the **IPAddresses** field are those which the device has been assigned, behind the NAT. But the **TranslatedIP** is the IP address of the NAT device.

TranslatedPort (2 bytes): The outbound TCP port for the device's SSTP connection to the server, as recognized by the presence server that is sending this notify message.

For example, if the device is behind a NAT device, the TCP port specified in the **ClientSSTPPort** field is the port on which the device listens, but the **TranslatedPort** is the outbound TCP port of the NAT device.

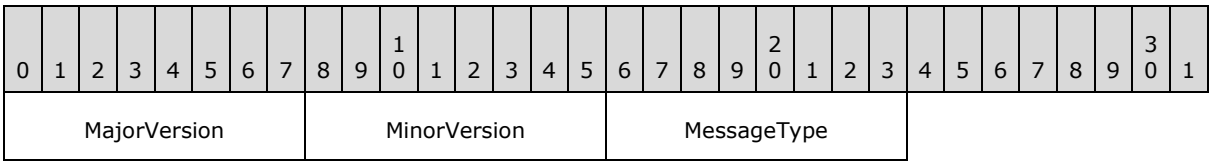
DPPSessionID (4 bytes): The identifier that the publishing client provided with the **Publish** message.

ClientPlatformVersion (variable): An ASCII string terminated by 0x00 that the publishing client provided with the **Publish** message.

2.2.9 Noop for WAN DPP 4.1

The **Noop** message can be sent from a client to a presence server. This message does nothing; the presence server discards Noop messages.

Noop fields are defined as shown in the following table.



MajorVersion (1 byte): Indicates the WAN DPP major version. This value MUST be 0x04.

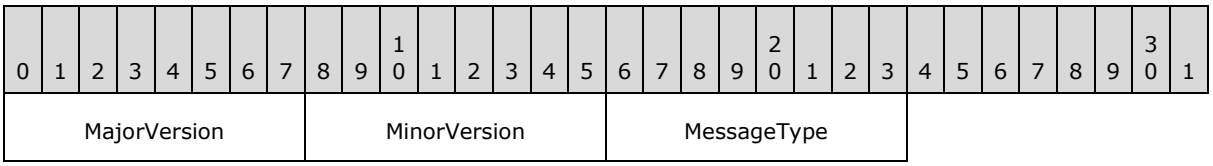
MinorVersion (1 byte): Indicates the WAN DPP minor version. This value MUST be 0x01.

MessageType (1 byte): Indicates a WAN DPP message type. For a **Noop** message, this field MUST be 0x04.

2.2.10 Noop for WAN DPP 5.0

The **Noop** message can be sent from a client to a presence server. This message does nothing; the presence server discards Noop messages.

Noop fields are defined as shown in the following table.



MajorVersion (1 byte): Indicates the WAN DPP major version. This value MUST be 0x05.

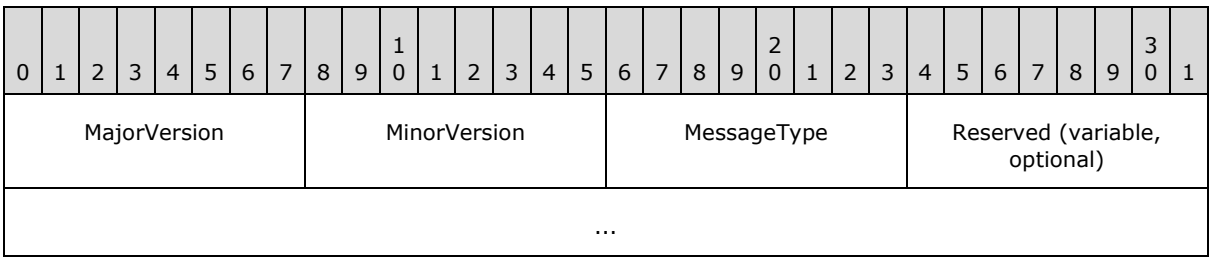
MinorVersion (1 byte): Indicates the WAN DPP minor version. This value MUST be 0x00.

MessageType (1 byte): Indicates a WAN DPP message type. For a **Noop** message, this field MUST be 0x04.

2.2.11 VersionRejected for WAN DPP 4.1

The **VersionRejected** message is sent from a presence server to a client. The server sends this message when it cannot process a WAN DPP message of a greater version than it supports.

VersionRejected fields are defined as shown in the following table.



MajorVersion (1 byte): The WAN DPP major version. This value MUST be 0x04.

MinorVersion (1 byte): The WAN DPP minor version. This value MAY<3> be 0x01.

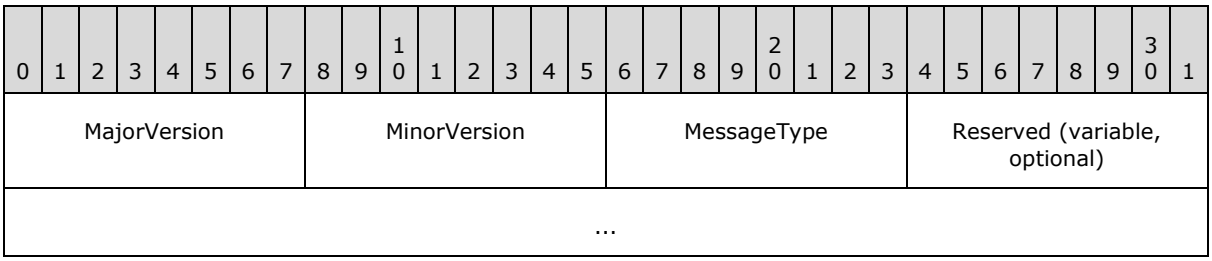
MessageType (1 byte): The WAN DPP message type. For a **VersionRejected** message, this field MUST be 0x06.

Reserved (variable, optional): A reserved field. This field MAY<4> be omitted.

2.2.12 VersionRejected for WAN DPP 5.0

The **VersionRejected** message is sent from a presence server to a client. The server sends this message when it cannot process a WAN DPP message of a greater version than it supports.

VersionRejected fields are defined as shown in the following table.



MajorVersion (1 byte): The WAN DPP major version. This value MUST be 0x05.

MinorVersion (1 byte): The WAN DPP minor version. This value MUST be 0x00.

MessageType (1 byte): The WAN DPP message type. For a **VersionRejected** message, this field MUST be 0x06.

Reserved (variable, optional): A reserved field. This field MAY<5> be omitted.

3 Protocol Details

3.1 Common 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.

Presence information is always associated with a client device as specified by a DeviceURL.

Presence information is defined by the following parameters:

Status: Indicates whether the device is online or offline.

IPAddresses: Contains the IP addresses on which the device listens for SSTP protocol messages.

ClientSSTPPort: Indicates the TCP port number on which the device listens for SSTP protocol messages.

TranslatedIP: Indicates the recognized IP address for the device as determined by the presence server.

TranslatedPort: Indicates the recognized TCP port number for the device as determined by the presence server.

DPPSessionID: Identifies an instance of a device's online status. This value is provided by the higher layer application.

ClientPlatformVersion: Indicates the version of the application running on the client device.

All devices implementing this protocol need to be aware of the version of WAN DPP that they are implementing.

DeviceDPPVersion: Specifies the version of the WAN DPP protocol that the local device is implementing. This variable MUST be either 4.1 or 5.0<6>.

Devices need to also be aware of the WAN DPP version being used on each of the open sessions that they have with other devices. They MUST maintain one instance of the following variable for each open session:

SessionDPPVersion: Specifies the version of the WAN DPP protocol that is being used over this session. This variable MUST be either 4.1 or 5.0.

3.1.2 Timers

None.

3.1.3 Initialization

The DeviceDPPVersion parameter needs to be set. If the version of SSTP being used as the transport is 1.5 then DeviceDPPVersion MUST be set to 4.1, and if the SSTP version is 1.6 then DeviceDPPVersion MUST be set to 5.0.

3.1.4 Higher-Layer Triggered Events

None.

3.1.5 Message Processing Events and Sequencing Rules

For any WAN DPP message received:

If the MajorVersion field of the message is equal to 4 and the DeviceDPPVersion is 4.1, the device MUST process the message. If the DeviceDPPVersion is 5.0 and the MajorVersion is equal to 5 the device also MUST process the message.

If the MajorVersion field of the message is equal to 3 or lower, the device MAY<7> ignore this message. If the MajorVersion field of the message is equal to 6 or greater clients MUST ignore this message and servers MUST reply with a VersionRejected message, as specified in 3.4.5.

Once the MajorVersion has been determined, the device identifies the message type and verifies that all required fields are present in the message. If any required field is not present, the device MUST ignore the message. If a required field contains an invalid value, the behavior is undefined.

Clients and servers participating in the WAN DPP protocol MUST ignore messages that exceed 4096 bytes.

3.1.6 Timer Events

None.

3.1.7 Other Local Events

3.1.7.1 Open WAN DPP Session

To open a WAN DPP session over SSTP a device MUST follow the procedure outlined in [MS-GRVSSTP] section 3.1.4.3, and use the parameters specified in section 2.1 of this document.

When opening a WAN DPP session, before sending any WAN DPP messages, the SessionDPPVersion parameter needs to be set. If the version of SSTP being used as the transport for this session is 1.5 then SessionDPPVersion MUST be set to 4.1, and if the SSTP version is 1.6 then the SessionDPPVersion MUST be set to 5.0.

3.2 Publishing Client Details

A publishing client's basic operation is to connect and publish to the presence server that it has been assigned to. A publishing client does the following:

- Opens an SSTP session to the presence server as specified in section 3.1.7.1.
- Sends, over the SSTP session, the Publish message that contains the client's presence information.
- Maintains the WAN DPP session as long as necessary to keep this presence information published and available to subscribing clients.

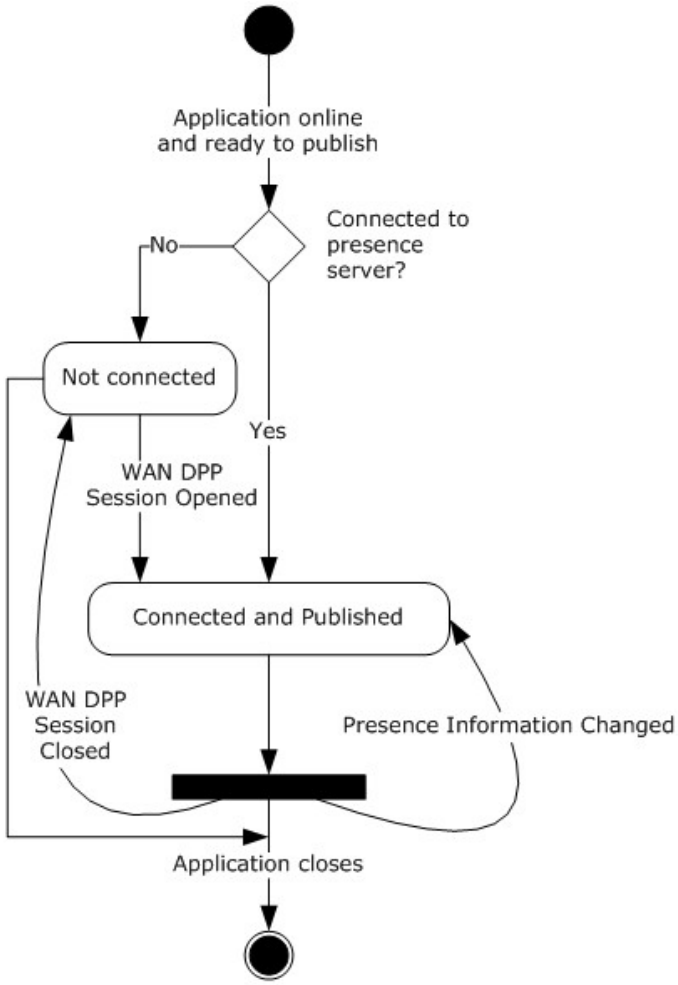


Figure 3: Publishing Client State

3.2.1 Abstract Data Model

This section describes a conceptual model of possible data organization that an implementation maintains to participate in this protocol. The described organization is provided to facilitate the explanation of how the protocol behaves. This document does not mandate that implementations adhere to this model as long as their external behavior is consistent with that described in this document.

The process of publishing presence information depends on the following parameters:

LocalDeviceURL: Identifier for the publishing client device. The publishing client stores its own device URL.

LocalPresenceServer: The name of the presence server assigned to the local device. The publishing client stores this information.

PublicationState: One of two values:

Not Connected - Indicates that the publishing client is not currently connected and its presence information is not published.

Connected and Published - Indicates that the client's presence information is published.

PresenceInformation: The device information and online/offline status of the device. The publishing client maintains presence information for itself and sends this information to the presence server via the Publish message.

3.2.2 Timers

None.

3.2.3 Initialization

A client application initializes in a state of not publishing presence information. A higher-layer triggered event starts the publishing process, as specified in section 3.2.4.1.

3.2.4 Higher-Layer Triggered Events

3.2.4.1 Application Requires its Presence Information Published

This event starts the publishing client state machine. The publishing client MUST do the following:

1. Determine its LocalDeviceURL and LocalPresenceServer.
2. Initialize its PublicationState to Not Connected.
3. Determine the ClientPlatformVersion field of its presence information which indicates the version of the software on the publishing client.
4. Initialize any application-specific mechanism necessary to determine IP address changes on the device.
5. Request a new DPPSessionID from the higher-layer application and store it in the presence information. This DPPSessionID MUST differ from any DPPSessionID used in any WAN DPP session between the client and server during the current SSTP connection.
6. If the PublicationState is Not Connected, the publishing client initiates a WAN DPP session to the presence server as specified in section 3.1.7.1.
7. Create a Publish message containing this presence information as specified in section 2.2.1 for WAN DPP version 4.1, or in section 2.2.2 for WAN DPP Version 5.0. The Status field of this presence information MUST be set to Online (0x80).
8. Send the Publish message to the presence server.
9. Set the PublicationState to Connected and Published.

A client that is both a publishing client and a subscribing client MAY<8> subscribe to device presence for its own device.

3.2.4.2 Application Closes

No action from the application is necessary because the operating system closes the application's TCP connections.

3.2.5 Message Processing Events and Sequencing Rules

3.2.5.1 Receiving a VersionRejected Message

When a publishing client receives a VersionRejected message from a server, it SHOULD<9> update its SessionDPPVersion state variable to be the same as or lower than the version of the message from the server.

After updating the SessionDPPVersion, the client SHOULD<10> send a Publish message to the server using the version of WAN DPP that SessionDPPVersion was just set to, as specified in section 3.2.4.1.

The client MUST ignore the Reserved field.

3.2.5.2 Receiving a WAN DPP message

A publishing client that is not also a subscribing client MUST ignore any WAN DPP messages other than the VersionRejected message.

If the client is also a subscribing client, then the received messages MUST be processed as specified in section 3.3.5.

3.2.6 Timer Events

None.

3.2.7 Other Local Events

3.2.7.1 WAN DPP Session Closed

When the WAN DPP session from the publishing client to the presence server is closed, the client MUST set the PublishingState to Not Connected and reestablish the session and re-publish its presence by following the procedure specified in section 3.2.4.1.

3.2.7.2 Presence Information Changed

A change in the list of IP addresses on a publishing client is an example of change in presence information. If the PublicationState is Connected and Published, the publishing client MUST send a Publish message with its new presence information, such as its new set of IP addresses, as specified in section 3.2.4.1, except that the DPPSessionID MUST be the DPPSessionID generated when the client first published.

3.3 Subscribing Client Details

A subscribing client requests notification of updates to the presence information of specified devices. The subscribing client MUST do the following:

1. Open a WAN DPP session to the presence server as specified in section 3.1.7.1. This is the assigned presence server of the device to which this client is subscribing.

2. Send a Subscribe message to the presence server requesting updates to presence information for specified devices.

3. Process Notify messages that contain presence information for specified devices.

4. Send an Unsubscribe message to the presence server or close the WAN DPP session when it no longer needs presence information updates.

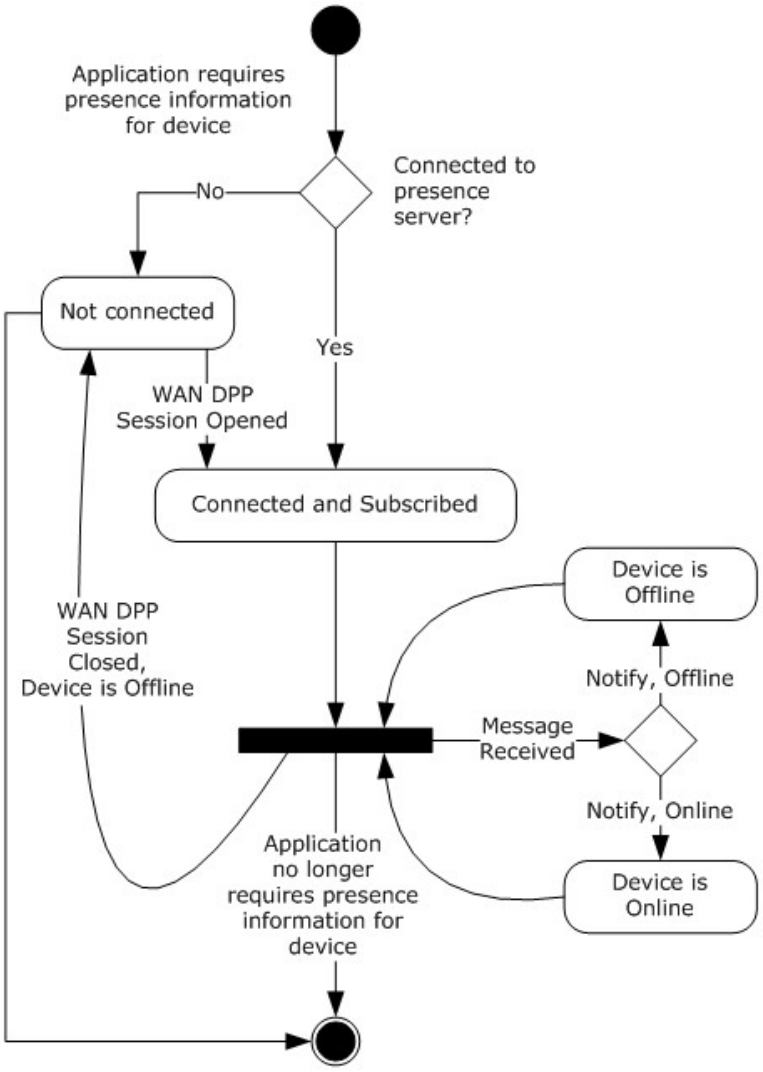


Figure 4: Subscribing Client State

3.3.1 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.

Subscribing to presence information depends on two sets of parameters, one for each target device and one for each presence server.

For each target device, a subscribing client maintains the following parameters:

DeviceURL: The identifier of the target device.

PresenceServerName: The name of the presence server for the device.

SubscriptionState: One of two values:

- Not Subscribed - Indicates that the client has not subscribed to the presence information for the target device.
- Subscribed - Indicates that the client has subscribed to the presence information for the target device.

SubscriptionID: An identifier that the subscribing client creates and associates with the active subscription for this device. This identifier is used to validate Notify messages received from the presence server.

PresenceInformation: The device information and online/offline status of the device being subscribed to.

In addition to the preceding parameters, a subscribing client maintains for each presence server a record consisting of the following parameters:

PresenceServerName: The name of the presence server.

ConnectionState: One of two values:

- Not Connected - Indicates that a session is not open to the presence server.
- Connected - Indicates that a session to the presence server is open.

DeviceURLs: A list of the DeviceURLs of the target devices that the subscribing client has subscribed to on this server.

3.3.2 Timers

None

3.3.3 Initialization

A subscribing client MUST assume that all other devices are offline at time of initialization.

3.3.4 Higher-Layer Triggered Events

3.3.4.1 Application Requires Presence Information for a Device

The client determines the presence server for the device and adds the device to the DeviceURLs list for this presence server.

If the ConnectionState for the target device's presence server is Not Connected, the subscribing client MUST open a session as specified in section 3.1.7.1 before sending a Subscribe. Once the steps in section 3.1.7.1 are finished follow the steps in 3.3.7.1 to set the Subscribing Client's local state variables for this connection.

Once the ConnectionState is Connected, the client sends a Subscribe message as follows:

1. Create a new SubscriptionID and stores this value. The SubscriptionID MUST be greater than zero and cannot be equal to the SubscriptionID of any other currently active subscription from this client to this server.

2. Create a Subscribe message (as specified in section 2.2.3 or 2.2.4) containing the pair of DeviceURL, SubscriptionID.

3. Send the Subscribe message to the presence server.

4. Set the SubscriptionState for the device to Subscribed.

3.3.4.2 Application No Longer Requires Presence Information for a Device

If the device is subscribed to presence information, as indicated by a Subscribed value for SubscriptionState, the subscribing client MUST do the following:

If SessionDPPVersion is 4.1:

1. Set the device status field within the PresenceInformation state variable for this device to offline.

2. Create an Unsubscribe message (as specified in section 2.2.5) containing the pair of DeviceURL, SubscriptionID. The SubscriptionID SHOULD<11> be the value stored previously when the Subscribe message was sent.

3. Send the Unsubscribe message to the presence server.

4. Set the SubscriptionState for the device to Not Subscribed.

If SessionDPPVersion is 5.0:

1. Set the device status field within the PresenceInformation state variable for this device to offline.

2. Create an Unsubscribe message (as specified in section 2.2.6) containing the SubscriptionID that was generated when the Subscribe message was sent.

3. Send the Unsubscribe message to the presence server.

4. Set the SubscriptionState for the device to Not Subscribed.

Additionally, the subscribing client MUST remove the device from the DeviceURLs list for the presence server.

3.3.5 Message Processing Events and Sequencing Rules

3.3.5.1 Receiving a Notify Message

The Notify message is sent from a presence server to a subscribing client to inform it of the most up-to-date presence information for a list of devices. A subscribing client MUST do the following:

1. Parse the Notify message. The message MUST be ignored if the MajorVersion is greater than 4 and the DeviceDPPVersion is 4.1, the MajorVersion is greater than 5 and the DeviceDPPVersion is 5.0, the size of the message is greater than 4096 bytes, or if any of the required fields are not present.

2. For each notification in the Notify message, the subscribing client MUST check the SubscriptionState.

1. If this is a WAN DPP 4.1 Notify message the value in the DeviceURL field MUST equal the DeviceURL of a target device as specified in section 3.3.1, the value in the SubscriptionID field MUST equal the SubscriptionID for that target device, and the SubscriptionState MUST be Subscribed. If any of these conditions is not met, this notification in the message MUST be ignored.

2. If this is a WAN DPP 5.0 Notify message the value in the DeviceURL field MUST be 0x00, the value in the SubscriptionID field MUST equal the SubscriptionID of a target device as specified in section 3.3.1, and the SubscriptionState for that device MUST be Subscribed. If any of these conditions is not met, this notification in the message MUST be ignored.

3. For each device identified by the Notify message, the subscribing client updates its stored presence information with the new information.

3.3.5.2 Receiving a VersionRejected Message

When a subscribing client receives a VersionRejected message from a server, it SHOULD<12> update its SessionDPPVersion state variable to be the same as or lower than the version of the message from the server.

If the VersionRejected message was received in response to a message that this client just sent, that message SHOULD<13> be re-sent, but with the new version information.

The client MUST ignore the Reserved field.

3.3.6 Timer Events

None.

3.3.7 Other Local Events

3.3.7.1 WAN DPP Session Opened

When a WAN DPP session from the client to the presence server opens, the subscribing client does the following:

1. If no record exists for the presence server, the subscribing client creates one.

2. Changes the ConnectionState for that presence server to Connected.

3. For each device stored in the DeviceURLs listed for this presence server, the subscribing client sends a Subscribe message as specified in 3.3.4.1.

3.3.7.2 WAN DPP Session Closed

When a WAN DPP session from the client to the presence server closes, the subscribing client does the following:

1. Changes the ConnectionState for that presence server to Not Connected.

2. For each device in that server's DeviceURLs list:

    1. The subscribing client sets the device status field within the PresenceInformation state variable for this device to offline.

2. The SubscriptionID is deleted and all further notifications for the pair of DeviceURL and SubscriptionID are ignored.

3. The SubscriptionState is set to Not Subscribed.

3.4 Presence Server Details

The presence server stores the presence information for publishing clients' devices and passes this information to subscribing clients. When subscribing clients open sessions and subscribe to presence information updates, the server sends Notify messages with the most current presence information that it has for those devices. The server then continues sending new Notify messages whenever presence information for those devices changes. If the subscribing client unsubscribes from presence information updates, the server stops sending Notify messages to that client.

3.4.1 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.

All presence servers maintain the following two collections of data structures:

- A subscribed-to/published device list which contains presence information for each device and a list of subscribing clients to that presence information.
- A subscribing client list which contains a list of the subscriptions that each subscribing client has made.

Each subscribed-to/published device record consists of the following parameters:

DeviceURL: The identifier of the device.

PresenceInformation: The current presence information for the device. Presence information is stored in fields whose values are initialized when a device record is first created. Field values are updated whenever a device publishes state changes; the Status is reset to its initial value when the device disconnects. Fields within this data type MUST be initialized as follows:

Field	Default Value
Status	Offline, 0x00
IPAddresses or IPAddressesV5	Empty list
ClientSSTPPort	0
TranslatedIP	"0.0.0.0"
TranslatedPort	0
DPPSessionID	0
ClientPlatformVersion	Empty string

SubscriberList: The server maintains a list of subscribing clients –The list identifies all the clients that subscribe to the presence information for the device.

Each subscribing client list entry contains the following fields:

DeviceURL: The identifier of the subscribing client.

SubscriptionList: A list of all the devices for which the client has requested presence information. Each entry in the list is made up of a pair of DeviceURL and SubscriptionID, where:

- **DeviceURL:** The URL for the device to which the subscribing client has subscribed.
- **SubscriptionID:** The SubscriptionID that the subscribing client provided in the Subscribe message for the device. This value is used in Notify messages to the subscribing client.

3.4.2 Timers

None.

3.4.3 Initialization

The server waits for incoming WAN DPP sessions to be opened from clients.

3.4.4 Higher-Layer Triggered Events

3.4.4.1 Request to Issue Notify Message

When the higher-layer triggers this event it will provide destination device information as well as the data for the Notify message itself.

1. Open an SSTP transport session to the destination device, as specified in section 3.1.7.1, if a session is not already open.
2. Create a Notify message that contains the data provided by the higher-layer.
3. Send the Notify message to the destination device<14>.

3.4.5 Message Processing Events and Sequencing Rules

The presence server MUST perform the following steps to validate and identify the incoming protocol messages:

1. If the message contains less than 3 bytes or greater than 4096 bytes, the message is invalid and the server MUST ignore the message.
2. If the MajorVersion is 3 or less, the server MAY<15> ignore the message.
3. If an SSTP transport session, as specified in section 2.1, to this client is not already open, the server MUST open one as specified in section 3.1.7.1 before sending any messages.
4. If the MajorVersion is greater than the major version in the DeviceDPPVersion variable, the server MUST send a VersionRejected message, using the version of WAN DPP set in DeviceDPPVersion, back to the client.

3.4.5.1 Receiving a Publish Message

The Publish message contains new presence information for a publishing client. The presence server MUST do the following:

1. Create a device record in the subscribe-to/published device list for the publishing client if a record does not already exist, determining the publishing client's DeviceURL from the SSTP session identifiers that were used to open the client to presence server SSTP session.

2. Update the contents of the associated PresenceInformation state variable for the publishing client based on the parameters provided in the Publish message.

3. Determine the source IP address and TCP port of the publishing client and stores these values in the TranslatedIP and TranslatedPort fields of the presence information.

4. Create and send a Notify message to each of the clients in the SubscriberList for the publishing client. For each client in the SubscriberList, the presence server MUST do the following:

    1. Create a Notify message that contains the SubscriptionID and, if the SessionDPPVersion is 4.1, the DeviceURL stored in the SubscriptionList for the subscribing client. The presence information fields are populated with the contents of the PresenceInformation state variable for the publishing client.

    2. Inform the higher-layer of a protocol initiated Request to Issue a Notify Message event (as specified in section 3.4.4.1), providing the subscribing client as the destination device and the just created Notify message.

3.4.5.2 Receiving a Subscribe Message

If the received message is a WAN DPP 5.0 Subscribe and the EndServerURLfield is not equal to 0x00 the server MUST ignore this message.

The Subscribe message includes the list of devices whose presence information is requested. When it receives a Subscribe message, a presence server MUST do the following:

1. Create a device record in the subscribing client list for the subscribing client if a record does not already exist. The subscribing client's DeviceURL is determined from the SSTP session identifiers that were used to open the client to presence server SSTP session.

2. Add each DeviceURL, SubscriptionID pair, specified in the Subscribe message, to the SubscriptionList that the server maintains for this subscribing client. If the SubscriptionList already contains this DeviceURL, the server MUST replace the stored SubscriptionID with the new one.

3. For each DeviceURL specified in the Subscribe message, the presence server MUST create a device record in the subscribed-to/published list if a record does not already exist, and add the subscribing client's DeviceURL to the SubscriberList for that DeviceURL if it is not already present.

4. If the device being subscribed to is online, the presence server MUST send a Notify message to the subscribing client, as specified in section 3.4.5.1, step 4.

The server MUST ignore the flags field.

3.4.5.3 Receiving an Unsubscribe Message

The Unsubscribe message contains a list of devices whose presence information the subscribing client no longer requires. When receiving an Unsubscribe message, the presence server MUST do the following:

1. Find the device record in the subscribing client list for the subscribing client. If no record exists, the Unsubscribe message MUST be ignored because this subscribing client has no active subscriptions.

2. For each Subscription identified by a device URL specified in a WAN DPP 4.1 Unsubscribe message or identified by a SubscriptionID in a WAN DPP 5.0 message, the presence server looks up the device record in the SubscriptionList. If the SubscriptionList does not include such a device, the presence server MUST ignore the request to unsubscribe for this device. If found, it MUST remove the device record from the SubscriptionList.

3. Find the device record in the subscribe-to/published list for each DeviceURL specified in the Unsubscribe message. If a record is found, the presence server removes the subscribing client's DeviceURL from the SubscriberList.

The server MUST ignore the flags field.

3.4.5.4 Receiving a Notify Message

A presence server MUST ignore any Notify messages that it receives.

3.4.5.5 Receiving a VersionRejected Message

A presence server MUST ignore any VersionRejected messages that it receives.

3.4.5.6 Receiving a Noop Message

A presence server MUST ignore any Noop message that it receives.

3.4.6 Timer Events

None.

3.4.7 Other Local Events

3.4.7.1 WAN DPP Session from Client Terminated

When an incoming WAN DPP session from a client is terminated, the presence server removes subscriptions and sets the Status for that client to offline (0x00).

The server MUST remove the subscriptions for all devices to whose presence information the client had subscribed. The server looks up the device record for the client in the subscribing client list. This device record contains the SubscriptionList containing active subscriptions from this client. The server looks up each target device in the subscribed-to/published device list and removes the subscribing client from the SubscriberList for the target device, and the server also removes all entries from the SubscriptionList for the subscribing client entry.

The presence server looks up the device record for the disconnected client in the subscribed-to/published device list. If the entry is found and the PresenceInformation state variable has a status of Online(0x80), the server MUST set that status to Offline(0x00), and the server MUST Inform the higher-layer of a protocol initiated Request to Issue a Notify Message event for each of the subscribing clients, as specified in section 3.4.5.1, step 4.

4 Protocol Examples

4.1 Separate Publisher and Subscriber Roles

This example involves a publishing client (Client A), a subscribing client (Client B), and a presence server. The three devices in this scenario perform the following tasks: Client A publishes its status to the presence server, Client B subscribes to Client A's status, and the presence server notifies Client B of Client A's presence information.

When Client A disconnects without publishing an update to its presence information, the server updates the presence information with a status of offline and notifies B. When Client B disconnects without sending an Unsubscribe messages for all subscriptions, the server removes all of Client B subscriptions.

While this scenario is not typical of a production environment, it highlights the client and server events involved in a simple WAN DPP message exchange. The following diagram illustrates this basic message exchange.

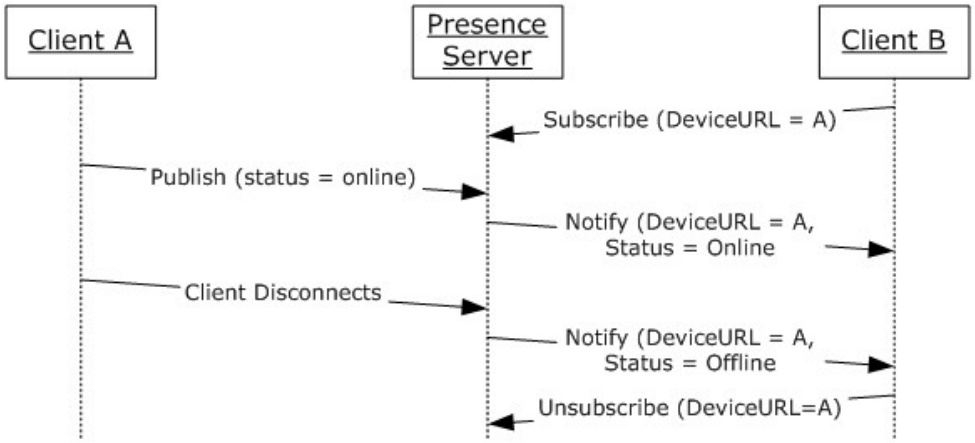


Figure 5: : Sample Message Exchange for Publishing Client, Subscribing Client, and Shared Presence Server

4.2 Simultaneous Publishing and Subscribing

This example illustrates the most common scenario for WAN DPP, where each device assumes the roles of a publishing client and a subscribing client simultaneously.

The two clients use the same presence server.

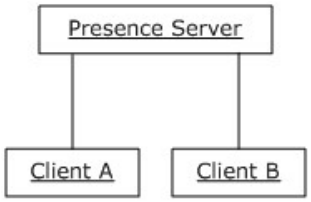


Figure 6: Two Clients Sharing a Presence Server

In this scenario, Client A and Client B publish their own device presence information to the server and subscribe to each other's published information. Clients can send Publish and Subscribe messages in any order.

Because Client B is offline when Client A subscribes to Client B's presence information, the presence server does not respond immediately to Client A's Subscribe message. The presence server will only send a Notify message when Client B comes online and publishes its presence information. In the meantime, Client A assumes that Client B is offline.

Client A sends Unsubscribe messages for all subscriptions and publishes its offline status, while Client B simply disconnects and the presence server behaves as if that Client B had sent Unsubscribe messages and published a status of offline.

The following diagram shows how Publish and Subscribe messages can occur in any sequence and how the server behaves appropriately, regardless of how a client ends a session.

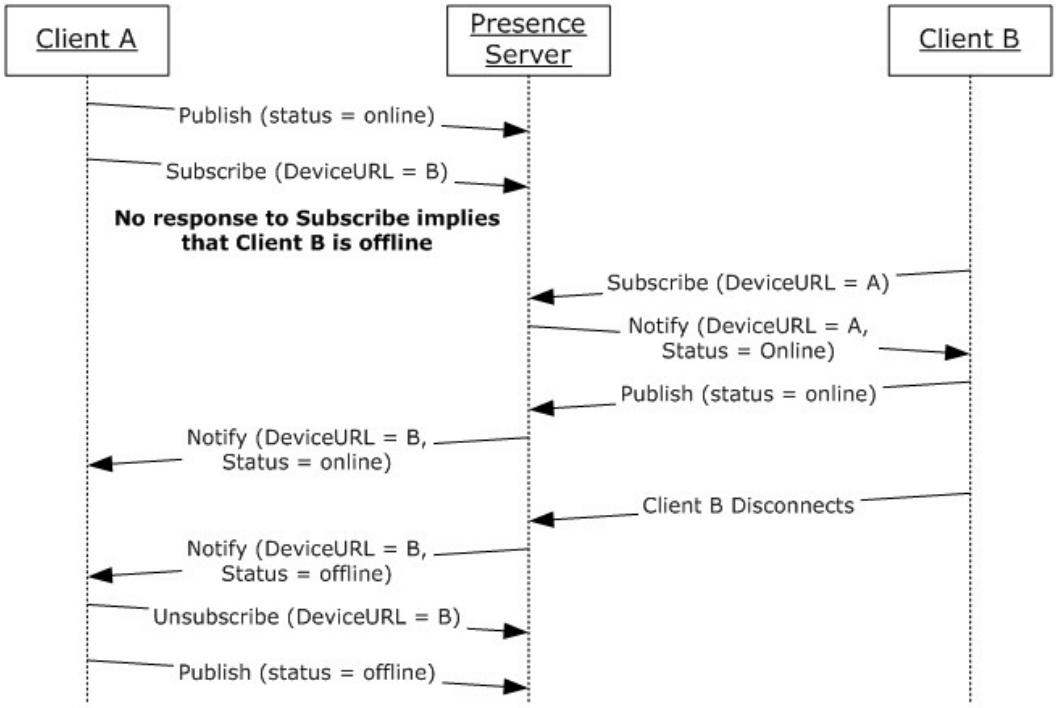


Figure 7: Sample Message Exchange for Two Clients Sharing a Presence Server

4.3 Clients Utilizing Separate Presence Servers Using WAN DPP 4.1

This example involves two WAN DPP clients that publish presence information and subscribe to each other's published information, as in the previous example. Unlike the previous example, in this case the two clients use two separate presence servers. This scenario illustrates how each client publishes device information to its assigned presence server and subscribes to another device's presence information about the presence server assigned to that device.

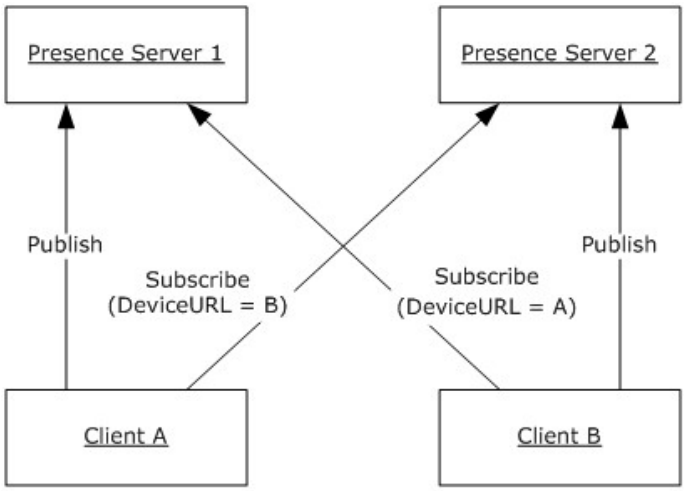


Figure 8: Two Clients with Different Presence Servers (WAN DPP 4.1)

The following diagram shows the message exchange among WAN DPP clients that publish and subscribe to different presence servers.

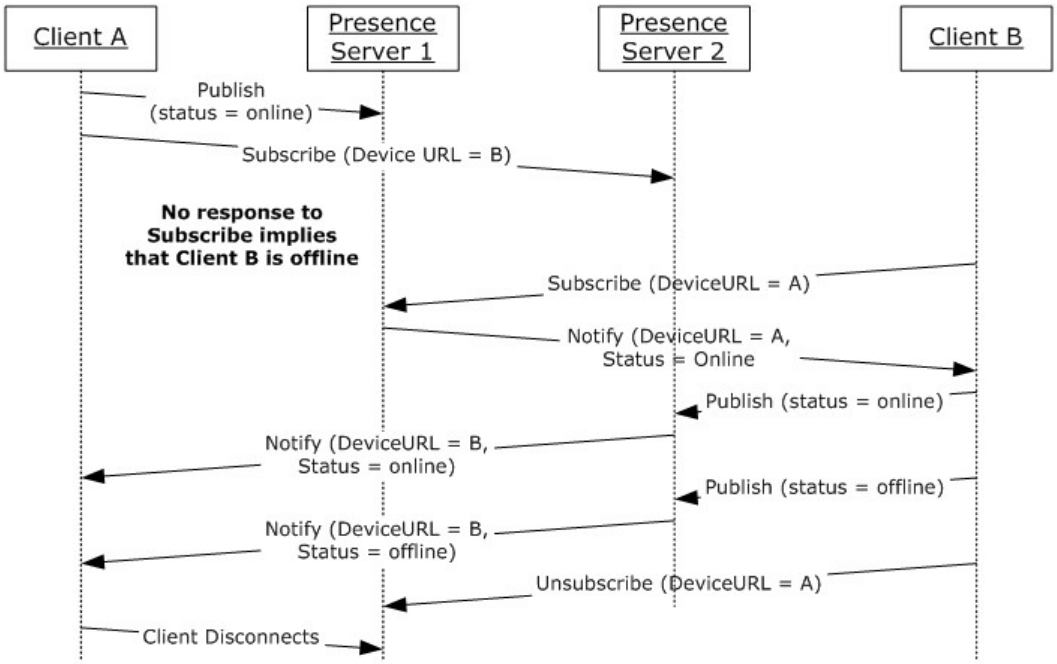


Figure 9: Sequence Diagram for Two Clients with Different Presence Servers (WAN DPP 4.1)

4.4 Messages

This section presents examples of the messages sent across the wire during WAN DPP exchanges.

4.4.1 Publish for WAN DPP 4.1

This is a valid message only when sent from a publishing client to a presence server.

The receiving presence server interprets the following sample message to mean that the publishing client is publishing the information that it is online and listening for commands on port 2492 at an IP address of 10.10.1.10. The server will store this information, and forward it on to subscribing clients that have subscribed to the presence of this client.

The following is a sample Publish message as encapsulated in an SSTP Data command:

```
0000 04 01 00 80 01 0a 01 0a 0a bc 09 92 55 b4 67 34 ............U.g4
0010 2c 32 2c 30 2c 32 36 32 33 00                   ,2,0,2623.
The bytes are as follows:
DPP41Data: DPP 41 Frame, MessageType : Publish
MajorVersion: 4 (0x04)
MinorVersion: 1 (0x01)
MessageType: 0 (0x00)
- Publish: Status: 0x80
Status: 128 (0x80)
NumberOfIPAddr: 1 (0x01)
IPAddresses: 10.10.1.10 (0x0A0A010A)
ClientSSTPPort: 2492 (0x09BC)
DPPSessionID: 1739871634 (0x67B45592)
ClientPlatformVersion: 4,2,0,2623
```

4.4.2 Subscribe for WAN DPP 4.1

This message is valid only when sent to a presence server.

The receiving presence server interprets the following sample message to mean that the subscribing client is subscribing to presence information about two publishing clients, with the DeviceURLs dpp:///jgnezs3gfkbykd6tnh2khrcnk2knh53dauidxj2 and dpp:///r9ya36rp6pyq2e4muc9d4nfg5kxf9jqd5wnqkha. The message associates those subscriptions with the subscription IDs 16 and 17.

The following is a sample Subscribe message encapsulated in an SSTP Data command:

```
0000 04 01 01 02 00 64 70 70 3a 2f 2f 2f 6a 67 6e 65 .....dpp:///jgne
0010 7a 73 33 67 66 6b 62 79 6b 64 36 74 6e 68 32 6b zs3gfkbykd6tnh2k
0020 68 72 63 6e 6b 32 6b 6e 68 35 33 64 61 75 69 64 hrcnk2knh53dauid
0030 78 6a 32 00 00 10 00 00 00 64 70 70 3a 2f 2f 2f xj2......dpp:///
0040 72 39 79 61 33 36 72 70 36 70 79 71 32 65 34 6d r9ya36rp6pyq2e4m
0050 75 63 39 64 34 6e 66 67 35 6b 78 66 39 6a 71 64 uc9d4nfg5kxf9jqd
0060 35 77 6e 71 6b 68 61 00 00 11 00 00 00          5wnqkha......
```

The bytes are as follows:

```
- DPP41Data: DPP 41 Frame, MessageType : Subscribe
MajorVersion: 4 (0x04)
MinorVersion: 1 (0x01)
MessageType: 1 (0x01)
- Subscribe: # of Devices: 2
NumberOfDevices: 2 (0x0002)
- SubscriptionData: SubscriptionID: 16, Flags: 0x00
```

```
DeviceURL: dpp:///jgnezs3gfkbykd6tnh2khrcnk2knh53dauidxj2
Flags: 0 (0x00)
SubscriptionID: 16 (0x00000010)
- SubscriptionData: SubscriptionID: 17, Flags: 0x00
DeviceURL: dpp:///r9ya36rp6pyq2e4muc9d4nfg5kxf9jqd5wnqkha
Flags: 0 (0x00)
SubscriptionID: 17 (0x00000011)
```

4.4.3 Unsubscribe for WAN DPP 4.1

This message is valid only when sent to a presence server.

The receiving presence server interprets the following sample message to mean that the subscribing client is unsubscribing from presence information about the publishing client with the DeviceURL

```
dpp:///r9ya36rp6pyq2e4muc9d4nfg5kxf9jqd5wnqkha.
```

The following is a sample Unsubscribe message as encapsulated in an SSTP Data command:

```
0000 04 01 02 01 00 64 70 70 3a 2f 2f 2f 72 39 79 61 .....dpp:///r9ya
0010 33 36 72 70 36 70 79 71 32 65 34 6d 75 63 39 64 36rp6pyq2e4muc9d
0020 34 6e 66 67 35 6b 78 66 39 6a 71 64 35 77 6e 71 4nfg5kxf9jqd5wnq
0030 6b 68 61 00 00 00 00 00 00                     kha......
```

The bytes are as follows:

```
- DPP41Data: DPP 41 Frame, MessageType : Unsubscribe
MajorVersion: 4 (0x04)
MinorVersion: 1 (0x01)
MessageType: 2 (0x02)
- Unsubscribe: # of Devices: 1
NumberOfDevices: 1 (0x0001)
- SubscriptionData: SubscriptionID: 0, Flags: 0x00
DeviceURL: dpp:///r9ya36rp6pyq2e4muc9d4nfg5kxf9jqd5wnqkha
Flags: 0 (0x00)
SubscriptionID: 0 (0x00000000)
```

4.4.4 Notify for WAN DPP 4.1

This is a valid message only when sent from a presence server to a device that has subscribed to presence information for the publishing client whose URL appears in the DeviceURL field.

The receiving subscribing client interprets the following sample message to mean that the sending presence server is notifying it that the publishing client is offline and listening for commands on port 2492 at an IP address of 10.10.1.10. After processing this message the subscribing client will be updated on the most recent presence information from the publishing client.

The following is a sample Notify message as encapsulated in an SSTP Data command:

```
0000 04 01 03 01 00 64 70 70 3a 2f 2f 2f 6a 67 6e 65 .....dpp:///jgne
0010 7a 73 33 67 66 6b 62 79 6b 64 36 74 6e 68 32 6b zs3gfkbykd6tnh2k
0020 68 72 63 6e 6b 32 6b 6e 68 35 33 64 61 75 69 64 hrcnk2knh53dauid
0030 78 6a 32 00 0b 00 00 00 00 01 0a 01 0a 0a bc 09 xj2.............
```

```
0040 0a 01 0a 0a 33 04 92 55 b4 67 34 2c 32 2c 30 2c ....3..U.g4,2,0,
0050 32 36 32 33 00                                  2623.
```

The bytes are as follows:

```
- DPP41Data: DPP 41 Frame, MessageType : Notify
MajorVersion: 4 (0x04)
MinorVersion: 1 (0x01)
MessageType: 3 (0x03)
- Notify: Notification Count: 1
NumberOfNotifications: 1 (0x0001)
- Notification: SubscriptionID: 0xb, Status: 0x00
DeviceURL: dpp:///jgnezs3gfkbykd6tnh2khrcnk2knh53dauidxj2
SubscriptionID: 11 (0x0000000B)
Status: 0 (0x00)
NumberOfIPAddr: 1 (0x01)
IPAddresses: 10.10.1.10 (0x0A0A010A)
ClientSSTPPort: 2492 (0x09BC)
TranslatedIP: 10.10.1.10 (0x0A0A010A)
TranslatedPort: 1075 (0x0433)
DPPSessionID: 1739871634 (0x67B45592)
ClientPlatformVersion: 4,2,0,2623
```

4.4.5 Publish for WAN DPP 5.0

This is a valid message only when sent from a publishing client to a presence server that supports WAN DPP 5.0.

The receiving presence server interprets the following sample message to mean that the publishing client is publishing the information that it is online and listening for commands on port 2492 at an IPv4 address of 10.10.1.10 and an IPv6 address of 2001:db8::1234:56ab. The server will store this information, and forward it on to subscribing clients that have subscribed to the presence of this client.

The following is a sample Publish message as encapsulated in an SSTP Data command:

```
0000  05 00 00 80 02 01 0a 01 0a 0a 02 10 20 01 0d b8  ............ ...
0010  00 00 00 00 00 00 00 00 12 34 56 ab bc 09 22 1b  .........4V...".
0020  f9 0b 31 34 2c 30 2c 30 2c 34 30 30 36 00        ..14,0,0,4006.
```

The bytes are as follows:

```
- DPP50Data: DPP 50 Frame, MessageType : Publish
   MajorVersion: 5 (0x05)
   MinorVersion: 0 (0x00)
   MessageType: 0 (0x00)
 - Publish: Status: 0x80
        Status: 128 (0x80)
      - IPAddresses: Count: 0x2
            NumberOfIPAddr: 2 (0x02)
            AddressType: 1 (0x01)
            IPv4Addr: 10.10.1.10 (0x0A0A010A)
            AddressType: 2 (0x02)
            IPv6Addr: 2001:DB8:0:0:0:0:1234:56AB
```

```
ClientSSTPPort: 2492 (0x09BC)
DPPSessionID: 200874786 (0x0BF91B22)
ClientPlatformVersion: 14,0,0,4006
```

4.4.6 Subscribe for WAN DPP 5.0

This message is valid only when sent to a presence server.

The receiving presence server interprets the following sample message to mean that the subscribing client is subscribing to presence information about a device with DeviceURL `dpp:///2ekxgnre72kmwj6eic3migktz62ezyzaxzg5asa` and to associated that subscription with SubscriptionID 7.

The following is a sample Subscribe message encapsulated in an SSTP Data command:

```
0000  05 00 01 01 00 64 70 70 3a 2f 2f 2f 32 65 6b 78  .....dpp:///2ekx
0010  67 6e 72 65 37 32 6b 6d 77 6a 36 65 69 63 33 6d  gnre72kmwj6eic3m
0020  69 67 6b 74 7a 36 32 65 7a 79 7a 61 78 7a 67 35  igktz62ezyzaxzg5
0030  61 73 61 00 00 00 07 00 00 00                    asa.......
```

The bytes are as follows:

```
- DPP50Data: DPP 50 Frame, MessageType : Subscribe
   MajorVersion: 5 (0x05)
   MinorVersion: 0 (0x00)
   MessageType: 1 (0x01)
 - Subscribe: # of Devices: 1
        NumberOfDevices: 1 (0x0001)
      - SubscriptionData: SubscriptionID: 7, Flags: 0x00
             DeviceURL: dpp:///2ekxgnre72kmwj6eic3migktz62ezyzaxzg5asa
             EndServerURL:
             Flags: 0 (0x00)
             SubscriptionID: 7 (0x00000007)
```

4.4.7 Unsubscribe for WAN DPP 5.0

This message is valid only when sent to a presence server.

The receiving presence server interprets the following sample message to mean that the subscribing client is unsubscribing from presence information about the previously subscribed to client with SubscriptionID 12.

The following is a sample Unsubscribe message as encapsulated in an SSTP Data command:

```
0000  05 00 02 01 00 00 00 00 0c 00 00 00              ............
```

The bytes are as follows:

```
- DPP50Data: DPP 50 Frame, MessageType : Unsubscribe
   MajorVersion: 5 (0x05)
   MinorVersion: 0 (0x00)
   MessageType: 2 (0x02)
 - Unsubscribe: # of Devices: 1
```

```
  NumberOfDevices: 1 (0x0001)
- SubscriptionData: SubscriptionID: 12, Flags: 0x00
      DeviceURL:
      EndServerURL:
      Flags: 0 (0x00)
      SubscriptionID: 12 (0x0000000C)
```

4.4.8 Notify for WAN DPP 5.0

This is a valid message only when sent from a presence server to a device that has subscribed to presence information for a publishing client that is identified by the unique SubscriptionID.

The receiving subscribing client interprets the following sample message to mean that the sending presence server is notifying it that the publishing client is offline and listening for commands on port 2492 at an IPv4 address of 10.10.1.10 and an IPv6 address of 2001:db8::1234:56ab. After processing this message the subscribing client will be updated on the most recent presence information from the publishing client.

The following is a sample Notify message as encapsulated in an SSTP Data command:

```
0000  05 00 03 01 00 00 00 09 00 00 00 00 02 01 0a 01  ................
0010  0a 0a 02 20 01 0d b8 00 00 00 00 00 00 00 00 12  ... ............
0020  34 56 ab bc 09 01 01 0a 01 0a 0a bc 09 22 1b f9  4V...........".. 
0030  0b 31 34 2c 30 2c 30 2c 34 30 30 36 00           .14,0,0,4006.
```

The bytes are as follows:

```
- DPP50Data: DPP 50 Frame, MessageType : Notify
   MajorVersion: 5 (0x05)
   MinorVersion: 0 (0x00)
   MessageType: 3 (0x03)
 - Notify: Notification Count: 1
      NumberOfNotifications: 1 (0x0001)
    - Notification: SubscriptionID: 0x9, Status: 0x00
         DeviceURL:
         EndServerURL:
         SubscriptionID: 9 (0x00000009)
         Status: 0 (0x00)
       - IPAddresses: Count: 0x2
            NumberOfIPAddr: 2 (0x02)
            AddressType: 1 (0x01)
            IPv4Addr: 10.10.1.10 (0x0A0A010A)
            AddressType: 2 (0x02)
            IPv6Addr: 2001:DB8:0:0:0:0:1234:56Ab
         ClientSSTPPort: 2492 (0x09BC)
       - TranslatedIP: Count: 0x1
            NumberOfIPAddr: 1 (0x01)
            AddressType: 1 (0x01)
            IPv4Addr: 10.10.1.10 (0x0A0A010A)
         TranslatedPort: 2492 (0x09BC)
         DPPSessionID: 200874786 (0x0BF91B22)
         ClientPlatformVersion: 14,0,0,4006
```

5 Security

5.1 Security Considerations for Implementers

The WAN DPP protocol does not have built-in security. The protocol does not provide any support for encryption, tamper detection, or authentication. As stated previously in this specification and shown in the examples in section 4, WAN DPP messages are sent as plaintext in SSTP Data commands.

The fact that WAN DPP messages are sent unencrypted exposes the protocol to the following types of security attacks:

- Eavesdropping - Neither recipient nor sender of a WAN DPP message can determine if the message was captured during transit and read by a third party.
- Tampering - Neither sender nor receiver of a WAN DPP message can determine if the message was intercepted during transfer and modified by a third party.
- Impersonation - A recipient of a WAN DPP message cannot authenticate the sender of a specific message.

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® Office 2010 suites
- Microsoft® Office Groove® 2007
- Microsoft® Office Groove® Server 2007
- Microsoft® Groove® Server 2010
- Microsoft® SharePoint® Workspace 2010

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: The Office Groove 2007 client or the SharePoint Workspace 2010 client is a WAN DPP publishing client and subscribing client. The Office Groove Server 2007 relay server or the Groove Server 2010 relay server is a WAN DPP presence server.

<2> Section 2.1: Office Groove Server 2007, and Groove Server 2010 do NOT send Notify messages across multiple SSTP Data messages. They will instead truncate the Notify message to ensure that it fits into the SSTP Data message size limit for a single message.

<3> Section 2.2.11: The Office Groove Server 2007 always sends the VersionRejected message using version 4.0 instead of 4.1.

<4> Section 2.2.11: Office Groove Server 2007 send non-0x00 values in this field, but any value is ignored.

<5> Section 2.2.12: Groove Server 2010 sends non-0x00 values in this field, but any value is ignored.

<6> Section 3.1.1: Office Groove 2007 and Office Groove Server 2007 use WAN DPP version 4.1, while SharePoint Workspace 2010 and Groove Server 2010 use version 5.0. Neither one allows the user to configure the version, and SharePoint Workspace 2010 and Groove Server 2010 support communication over either version for downward compatibility.

<7> Section 3.1.5: Office Groove 2007, SharePoint Workspace 2010, Office Groove Server 2007, and Groove Server 2010 process WAN DPP messages with versions prior to 4.1. Implementers of the protocol need not process any version prior to 4.1.

<8> Section 3.2.4.1: Office Groove 2007 and SharePoint Workspace 2010 subscribe to their own presence information to discover their public addresses. The server responds with the same information that the client publishes, but with the addition of the TranslatedIP and TranslatedPort information.

<9> Section 3.2.5.1: Office Groove 2007 ignores the VersionRejected message because it uses the SSTP protocol version to determine the correct WAN DPP version to use.

<10> Section 3.2.5.1: Office Groove 2007 ignores the VersionRejected message because it uses the SSTP protocol version to determine the correct WAN DPP version to use.

<11> Section 3.3.4.2: Office Groove 2007 always sends Unsubscribe messages with a SubscriptionID of zero.

<12> Section 3.3.5.2: Office Groove 2007 ignores the VersionRejected message because it uses the SSTP protocol version to determine the correct WAN DPP version to use.

<13> Section 3.3.5.2: Office Groove 2007 ignores the VersionRejected message because it uses the SSTP protocol version to determine the correct WAN DPP version to use.

<14> Section 3.4.4.1: Office Groove Server 2007 and Groove Server 2010 do not always send a Notify message for every request to issue a Notify Message event. Under some circumstance such as heavy server load, instead of immediately sending Notify messages, Office Groove Server 2007 and Groove Server 2010 aggregate presence changes and send the most up-to-date information after up to 2 seconds. Consequently, if multiple state transitions happen within the aggregation window, some of the intermediate states are skipped in the notification sequence.

However Office Groove 2007 and SharePoint Workspace 2010 require presence updates when a communicating client's state changes from online to offline and back to online within the aggregation window that Office Groove Server 2007 and Groove Server 2010 use to aggregate presence changes. To determine this condition, a subscribing Office Groove 2007 client or SharePoint Workspace 2010 client uses DPPSessionIDs (identifiers provided by publishing clients in Publish messages and distributed to subscribing clients in Notify messages). Office Groove Server 2007 and Groove Server 2010 pass published presence data to subscribing clients unaltered, sending it in a Notify message that also includes the published device's DPPSessionID. Office Groove 2007 and SharePoint Workspace 2010 generate a new DPPSessionID for each connection with the presence server and detect the online-offline-online condition as three changes in state.

<15> Section 3.4.5: Office Groove Server 2007 and Groove Server 2010 process versions of WAN DPP prior to 4.1. Implementers of the protocol need not process any version prior to 4.1.

7 Change Tracking

This section identifies changes that were made to the [MS-GRVWDPP] protocol document between the June 2011 and January 2012 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
2.1 Transport	Clarified product client and server roles.	N	New product behavior note added.
3.2.4.1 Application Requires its Presence Information Published	Clarified use of presence information by clients.	N	Product behavior note updated.
3.2.5.1 Receiving a VersionRejected Message	Clarified client use of the "VersionRejected" message.	N	New product behavior note added.
3.3.5.2 Receiving a VersionRejected Message	Clarified client use of the "VersionRejected" message.	N	New product behavior note added.
3.4.1 Abstract Data Model	Clarified server use of "SubscriberList".	N	Content updated.
3.4.4.1 Request to Issue Notify Message	Clarified server use of aggregation window.	N	Product behavior note updated.
3.4.5.5 Receiving a VersionRejected Message	Added section.	Y	New content added.

8 Index

T

U

V