[MS-ABS]:

Address Book File Structure

Intellectual Property Rights Notice for Open Specifications Documentation

- **Technical Documentation.** Microsoft publishes Open Specifications documentation for protocols, file formats, languages, standards as well as overviews of the interaction among each of these technologies.
- **Copyrights**. This documentation is covered by Microsoft copyrights. Regardless of any other terms that are contained in the terms of use for the Microsoft website that hosts this documentation, you may make copies of it in order to develop implementations of the technologies described in the Open Specifications and may distribute portions of it in your implementations using these technologies or your documentation as necessary to properly document the implementation. You may also distribute in your implementation, with or without modification, any schema, IDL's, or code samples that are included in the documentation. This permission also applies to any documents that are referenced in the Open Specifications.
- **No Trade Secrets**. Microsoft does not claim any trade secret rights in this documentation.
- **Patents**. Microsoft has patents that may cover your implementations of the technologies described in the Open Specifications. Neither this notice nor Microsoft's delivery of the documentation grants any licenses under those or any other Microsoft patents. However, a given Open Specification may be covered by Microsoft Open Specification Promise or the Community Promise. If you would prefer a written license, or if the technologies described in the Open Specifications are not covered by the Open Specifications Promise or Community Promise, as applicable, patent licenses are available by contacting iplg@microsoft.com.
- **Trademarks**. The names of companies and products contained in this documentation may be covered by trademarks or similar intellectual property rights. This notice does not grant any licenses under those rights. For a list of Microsoft trademarks, visit www.microsoft.com/trademarks.
- **Fictitious Names**. The example companies, organizations, products, domain names, e-mail addresses, logos, people, places, and events depicted in this documentation are fictitious. No association with any real company, organization, product, domain name, email address, logo, person, place, or event is intended or should be inferred.

Reservation of Rights. All other rights are reserved, and this notice does not grant any rights other than specifically described above, whether by implication, estoppel, or otherwise.

Tools. The Open Specifications do not require the use of Microsoft programming tools or programming environments in order for you to develop an implementation. If you have access to Microsoft programming tools and environments you are free to take advantage of them. Certain Open Specifications are intended for use in conjunction with publicly available standard specifications and network programming art, and assumes that the reader either is familiar with the aforementioned material or has immediate access to it.

Revision Summary

Date	Revision History	Revision Class	Comments
4/4/2008	0.1	Major	Initial Availability
4/25/2008	0.2	Major	Revised and edited the technical content
6/27/2008	1.0	Major	Revised and edited the technical content
8/15/2008	1.01	Editorial	Edited the technical content
12/12/2008	2.0	Major	Revised and edited the technical content
2/13/2009	2.01	Editorial	Edited the technical content
3/13/2009	2.02	Editorial	Edited the technical content
7/13/2009	2.03	Major	Revised and edited the technical content
8/28/2009	2.04	Editorial	Revised and edited the technical content
11/6/2009	2.05	Editorial	Revised and edited the technical content
2/19/2010	2.06	Editorial	Revised and edited the technical content
3/31/2010	2.07	Major	Updated and revised the technical content
4/30/2010	2.08	Editorial	Revised and edited the technical content
6/7/2010	2.09	Editorial	Revised and edited the technical content
6/29/2010	2.10	Editorial	Changed language and formatting in the technical content.
7/23/2010	2.10	No Change	No changes to the meaning, language, or formatting of the technical content.
9/27/2010	3.0	Major	Significantly changed the technical content.
11/15/2010	3.0	No Change	No changes to the meaning, language, or formatting of the technical content.
12/17/2010	3.0	No Change	No changes to the meaning, language, or formatting of the technical content.
3/18/2011	3.0	No Change	No changes to the meaning, language, or formatting of the technical content.
6/10/2011	3.0	No Change	No changes to the meaning, language, or formatting of the technical content.
1/20/2012	3.1	Minor	Clarified the meaning of the technical content.
4/11/2012	3.1	No Change	No changes to the meaning, language, or formatting of the technical content.
7/16/2012	3.1	No Change	No changes to the meaning, language, or formatting of the technical content.
10/8/2012	3.2	Minor	Clarified the meaning of the technical content.
2/11/2013	3.2	No Change	No changes to the meaning, language, or formatting of the technical content.

Date	Revision History	Revision Class	Comments
7/30/2013	3.3	Minor	Clarified the meaning of the technical content.
11/18/2013	3.3	No Change	No changes to the meaning, language, or formatting of the technical content.
2/10/2014	3.3.1	Editorial	Changed language and formatting in the technical content.
4/30/2014	3.3.1	No Change	No changes to the meaning, language, or formatting of the technical content.
7/31/2014	3.3.1	No Change	No changes to the meaning, language, or formatting of the technical content.
10/30/2014	3.3.1	No Change	No changes to the meaning, language, or formatting of the technical content.
3/30/2015	4.0	Major	Significantly changed the technical content.

Table of Contents

1 Introduction

The Address Book File Structure describes the format of the **address book files** that are produced daily by the **Address Book Server (ABS)** and accessed by clients to get information about users, **contacts** and **group objects** stored in **Active Directory**. Clients can use the contents of these address book files to provide an incremental search capability against the **users**, **contacts** and **groups** that were stored in Active Directory at the time the address book files were captured. Clients can also use this to provide reverse number lookup of incoming **Voice over IP (VoIP)** calls.

Sections 1.7 and 2 of this specification are normative and can contain the terms MAY, SHOULD, MUST, MUST NOT, and SHOULD NOT as defined in [RFC2119]. All other sections and examples in this specification are informative.

1.1 Glossary

The following terms are specific to this document:

Active Directory: A general-purpose network directory service. **Active Directory** also refers to the Windows implementation of a directory service. **Active Directory** stores information about a variety of objects in the network. Importantly, user accounts, computer accounts, groups, and all related credential information used by the Windows implementation of Kerberos are stored in **Active Directory**. **Active Directory** is either deployed as **Active Directory Domain Services (AD DS)** or Active Directory Lightweight Directory Services (AD LDS). [MS-ADTS] describes both forms. For more information, see [MS-AUTHSOD] section 1.1.1.5.2, Lightweight Directory Access Protocol (LDAP) versions 2 and 3, Kerberos, and DNS.

Active Directory Domain Services (AD DS): A directory service (DS) implemented by a domain controller (DC). The DS provides a data store for objects that is distributed across multiple DCs. The DCs interoperate as peers to ensure that a local change to an object replicates correctly across DCs. For more information, see [MS-AUTHSOD] section 1.1.1.5.2 and [MS-ADTS]. For information about product versions, see [MS-ADTS] section 1. See also **Active Directory**.

address book contact: A user, **contact**, or group object that is obtained from Active Directory Domain Services (AD DS), including a subset of the AD DS attributes that are associated with the object, and is stored in an address book file.

address book file: A file that contains a set of **address book contact** records.

Address Book Server (ABS): A component that produces **address book files** on a daily basis.

American National Standards Institute (ANSI) character set: A character set (1) defined by a code page approved by the American National Standards Institute (ANSI). The term "ANSI" as used to signify Windows code pages is a historical reference and a misnomer that persists in the Windows community. The source of this misnomer stems from the fact that the Windows code page 1252 was originally based on an ANSI draft, which became International Organization for Standardization (ISO) Standard 8859-1 [ISO/IEC-8859-1]. In Windows, the ANSI character set can be any of the following code pages: 1252, 1250, 1251, 1253, 1254, 1255, 1256, 1257, 1258, 874, 932, 936, 949, or 950. For example, "ANSI application" is usually a reference to a non-**Unicode** or code-page-based application. Therefore, "ANSI character set" is often misused to refer to one of the character sets defined by a Windows code page that can be used as an active system code page; for example, character sets defined by code page 1252 or character sets defined by code page 950. Windows is now based on **Unicode**, so the use of ANSI character sets is strongly discouraged unless they are used to interoperate with legacy applications or legacy data.

attribute: (A specialization of the previous definition.) An identifier for a single or multivalued data element that is associated with a directory object. An object consists of its **attributes** and their values. For example, cn (common name), street (street address), and mail (email addresses)

can all be **attributes** of a **user object**. An **attribute's** schema, including the syntax of its values, is defined in an attributeSchema object.

big-endian: Multiple-byte values that are byte-ordered with the most significant byte stored in the memory location with the lowest address.

contact: A person, company, or other entity that is stored in a directory and is associated with one or more unique identifiers and **attributes**, such as an Internet message address or login name.

cryptographic hash function: A function that maps an input of any length to a short output bit string of fixed length, such that finding an input that maps to a particular bit string of the correct output length, or even finding two inputs that map to the same output bit string, is computationally infeasible. For more information, see [SCHNEIER] chapters 2 and 18.

cyclic redundancy check (CRC): An algorithm used to produce a checksum (a small, fixed number of bits) against a block of data, such as a packet of network traffic or a block of a computer file. The CRC is used to detect errors after transmission or storage. A CRC is designed to catch random errors, as opposed to intentional errors. If errors might be introduced by a motivated and intelligent adversary, a **cryptographic hash function** should be used instead.

delta address book file: An **address book file** that contains only the differences between two complete address book files. Differences can include changed values, added objects, and deleted objects.

distinguished name (DN): In the **Active Directory** directory service, the unique identifier of an object in **Active Directory**, as described in [MS-ADTS] and [RFC2251].

full address book file: An **address book file** that contains a complete set of the **address book contacts** that existed when the file was generated by the user, contact, and groups objects in AD DS.

globally unique identifier (GUID): A term used interchangeably with universally unique identifier (UUID) in Microsoft protocol technical documents (TDs). Interchanging the usage of these terms does not imply or require a specific algorithm or mechanism to generate the value. Specifically, the use of this term does not imply or require that the algorithms specified in [RFC4122] or [C706] must be used for generating the **GUID**. See also universally unique identifier (UUID).

group object: In **Active Directory**, a **group object** has an object class group. A group has a forward link attribute member; the values of this **attribute** either represent elements of the group (for example, objects of class user or computer) or subsets of the group (objects of class group). The representation of group subsets is called "nested group membership". The back link attribute memberOf enables navigation from group members to the groups containing them. Some groups represent groups of security principals and some do not and are, for instance, used to represent email distribution lists.

hash: A fixed-size result that is obtained by applying a one-way mathematical function, which is sometimes referred to as a hash algorithm, to an arbitrary amount of data. If the input data changes, the hash also changes. The hash can be used in many operations, including authentication (2) and digital signing.

hash code: See **hash**.

in-band provisioning: A process in which a protocol client obtains configuration information from a protocol server.

little-endian: Multiple-byte values that are byte-ordered with the least significant byte stored in the memory location with the lowest address.

Session Initiation Protocol (SIP): An application-layer control (signaling) protocol for creating, modifying, and terminating sessions with one or more participants. **SIP** is defined in [RFC3261].

Unicode: A character encoding standard developed by the Unicode Consortium that represents almost all of the written languages of the world. The **Unicode** standard [UNICODE5.0.0/2007] provides three forms (UTF-8, UTF-16, and UTF-32) and seven schemes (UTF-8, UTF-16, UTF-16 BE, UTF-16 LE, UTF-32, UTF-32 LE, and UTF-32 BE).

user object: An object of class user. A user object is a security principal object; the principal is a person or service entity running on the computer. The shared secret allows the person or service entity to authenticate itself, as described in ([MS-AUTHSOD] section 1.1.1.1).

UTF-8: A byte-oriented standard for encoding Unicode characters, defined in the Unicode standard. Unless specified otherwise, this term refers to the UTF-8 encoding form specified in [UNICODE5.0.0/2007] section 3.9.

Voice over IP (VoIP): The use of the Internet Protocol (IP) for transmitting voice communications. VoIP delivers digitized audio in packet form and can be used to transmit over intranets, extranets, and the Internet.

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

1.2 References

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.

[E164] ITU-T, "The International Public Telecommunication Numbering Plan", Recommendation E.164, February 2005, http://www.itu.int/rec/T-REC-E.164/e

Note There is a charge to download the specification.

[MC-RegEx] Microsoft Corporation, "Regular Expression Language Elements", http://msdn.microsoft.com/en-us/library/az24scfc(VS.80).aspx

[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

[UASDC] Ziv, J. and Lempel, A., "A Universal Algorithm for Sequential Data Compression", May 1977, http://www.cs.duke.edu/courses/spring03/cps296.5/papers/ziv_lempel_1977_universal_algorithm.pdf

[UNICODE] The Unicode Consortium, "The Unicode Consortium Home Page", 2006, http://www.unicode.org/

1.3 Structure Overview (Synopsis)

Address book files are produced daily by the Address Book Server. These files represent a daily snapshot of the **user object**, contact and group object in the **Active Directory Domain Services (AD DS)**, represented as a set of **address book contacts** in the address book files. The address book files are stored in a file system folder that contains the most recent N days of address book files, where N is greater than 0. Client applications can schedule daily downloads of the address book file for

local, incremental searching of users, contacts, and groups based on various fields in the file. Client applications can also use the data in the address book file to perform reverse number lookup for incoming calls from other clients.

Clients access the address book files by using the HTTP GET method. Clients construct the URL of the latest address book file by concatenating the base URL they receive as part of **in-band provisioning** with the address book file name constructed from the date. The file name is constructed as follows:

F-XXXX.lsabs

D-XXXX-YYYY.lsabs

C-XXXX-YYYY.lsabs<1>

F-XXXX.dabs

D-XXXX-YYYY.dabs

Where:

- XXXX and YYYY are 4-digit hexadecimal values that represent dates as the 0-based number of days after January 1, 2001, 00:00:00 UTC. For example, F-0A8C.lsabs would be the address book file for Saturday, May 24, 2008.
- File names that begin with F-XXXX are called full files and contain all user object, contact and group objects that exist on a given day.
- Files that begin with D-XXXX-YYYY are called delta files and represent the difference between two full files (F-XXXX.ext and F-YYYY.ext), where XXXX is less than YYYY. A delta file contains new and changed user object, contact and group objects, as well as deleted user objects, contact and group objects.
- Files that begin with C-XXXX-YYYY are called compact delta files and represent the difference between two full files (F-XXXX.lsabs and F-YYYY.lsabs), where XXXX is less than YYYY. A compact delta file contains new and deleted user objects, contacts (3), and group objects (2), as well as the specific attribute changes for changed user objects, contacts (3), and group objects (2).<2>
- Files with the .lsabs extension contain, for each user object, contacts, and group objects, all **attributes** that would be of interest to an end user.
- Files with the .dabs extension contain a minimal subset of attributes that would be useful to a physical device with limited memory, such as a **Session Initiation Protocol (SIP)** phone client. Also, group objects (2) are never included in device-specific address book files, so setting this flag on the **groupType** attribute will have no effect.

1.3.1 Address Book Server Configuration

The Address Book Server relies on a number of settings to control what goes into address book files. At a minimum, each address book file includes the following configuration settings:

ABF_ATTRIBUTES: Controls which attributes for each **user**, **contact** and **group** are examined and how they are processed by the Address Book Server. A portion of this information is also used by client programs when interpreting downloaded address book files.

ABF_NORMALIZATION_RULES: Controls how the Address Book Server normalizes phone number attribute values. Normalization is the process of taking an arbitrarily formatted phone number and converting it into a dial string, as described in [E164]. These normalization rules are also stored in the address book files for use by clients when normalizing numbers outside of

the address book, such as from personal contacts repository or a phone number entered manually through the client UI.

UseNormalizationRules: Controls whether the Address Book Server puts normalized phone numbers in the address book file or leaves phone numbers as they are and lets the client perform the normalization if needed.

How these configuration settings are stored and accessed is at the discretion of the system implementer and is outside the scope of this document.

1.3.2 Compressed File Format

Each address book file is compressed using the proprietary compression algorithm described in section 5. The file is stored on disk in compressed form and read over the wire using the HTTP GET method by client software. The client software then decompresses the file locally. The format of the compressed address book file is one or more COMPRESSED_BLOCK records. Each COMPRESSED_BLOCK record consists of a COMPRESSED_BLOCK_HEADER followed by the bytes for that COMPRESSED_BLOCK. The decompression algorithm reads each block and decompresses it. The concatenation of all the decompressed blocks is the decompressed address book file.

1.3.3 Decompressed File Format

After an address book file is decompressed, it can be examined. For full files, the overall layout is as follows:

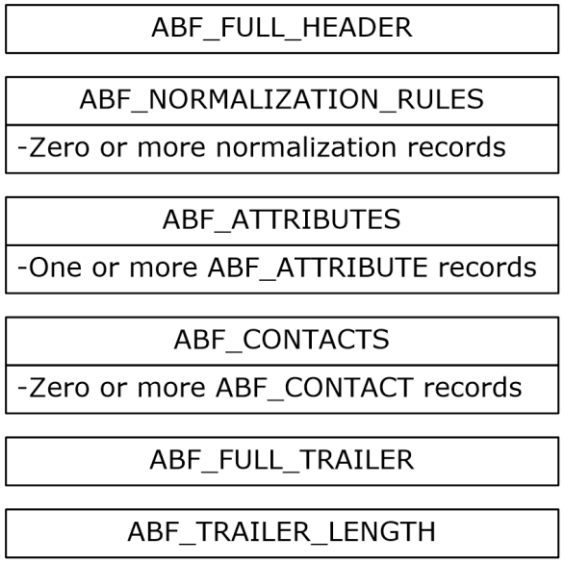


Figure 1: Address book full file decompression layout

For delta files, the overall layout is as follows:

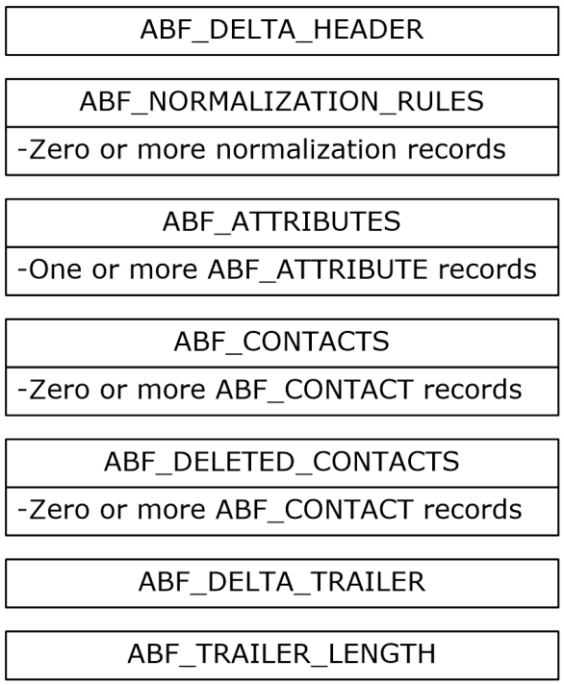


Figure 2: Address book full file decompression layout

For compact delta files, the overall layout is as follows:<3>

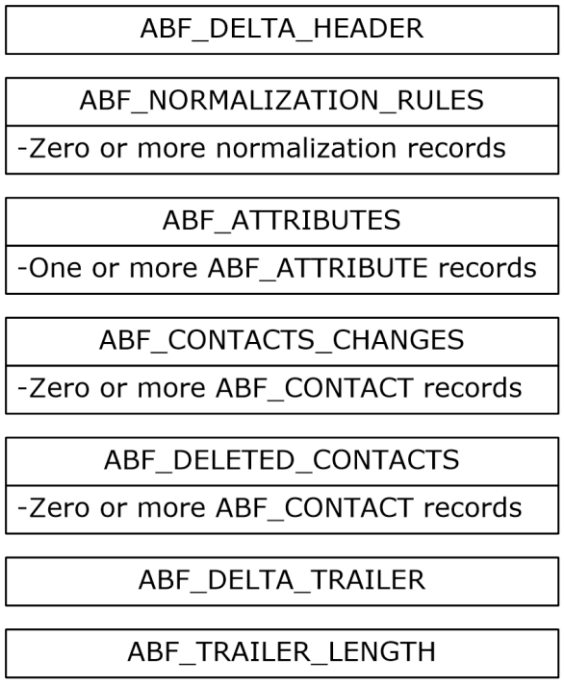


Figure 3: Address book compact delta file compression layout

1.3.4 Byte Ordering

Data in **address book file** records are stored in **little-endian** format.

Some computer architectures number bytes in a binary word from left to right, which is referred to as **big-endian**. The bit diagram for this documentation is big-endian. Other architectures number the bytes in a binary word from right to left, which is referred to as little-endian. The underlying file format enumerations, objects and records are little-endian.

Using the big-endian and little-endian methods, the number 0x12345678 would be stored as shown in the following table.

Byte order	Byte 0	Byte 1	Byte 2	Byte 3
big-endian	0x12	0x34	0x56	0x78
little-endian	0x78	0x56	0x34	0x12

1.4 Relationship to Protocols and Other Structures

None.

1.5 Applicability Statement

Files that adhere to the **address book file** structure are suitable for use by clients listed in section 8.

1.6 Versioning and Localization

This document covers versioning issues in the following areas:

- **Version:** This document specifies the first version of the **address book file** structure. The **ABF_TRAILER_LENGTH** structure provides support for adding more information in a way that does not affect older clients of the file structure.
- **Localization:** This structure defines no locale-specific processes or data. All strings are encoded in **Unicode UTF-8** [UNICODE] format.

1.7 Vendor-Extensible Fields

None.

2 Structures

The following sections specify various types of address book file records and enumerations.

Note: All character strings specified in this section MUST be encoded in Unicode UTF-8 format.

2.1 Address Book File Enumeration Tables

2.1.1 ABF_ATTRIBUTE_FLAGS Enumeration

The **ABF_ATTRIBUTE_FLAGS** enumeration defines the bit values stored in the Flags field of the **ABF_ATTRIBUTE** structure.

```
typedef enum
{
  ABF_ATTRIBUTE_FLAGS_TYPE_MASK        = 0x000000FF,
  ABF_ATTRIBUTE_FLAGS_RESERVED         = 0x00000100,
  ABF_ATTRIBUTE_FLAGS_EMAIL            = 0x00000400,
  ABF_ATTRIBUTE_FLAGS_REQUIRED         = 0x00000800,
  ABF_ATTRIBUTE_FLAGS_NC_MASK          = 0x00003000,
  ABF_ATTRIBUTE_FLAGS_EXCLUDE          = 0x00004000,
  ABF_ATTRIBUTE_FLAGS_INCLUDE          = 0x00008000,
  ABF_ATTRIBUTE_FLAGS_GROUPTYPE        = 0x00010000,
  ABF_ATTRIBUTE_FLAGS_DEVICE           = 0x00020000,
  ABF_ATTRIBUTE_FLAGS_UNUSED           = 0x00FC0200,
  ABF_ATTRIBUTE_FLAGS_CLIENT_MASK      = 0xFF000000,
} ABF_ATTRIBUTE_FLAGS;
```

ABF_ATTRIBUTE_FLAGS_TYPE_MASK: Used to mask out the value type of an Active Directory attribute. Doing a bitwise AND of this value with the **Flags** field of the **ABF_ATTRIBUTE** will yield a value in the **ABF_ATTRIBUTE_TYPE** enumeration.

ABF_ATTRIBUTE_FLAGS_RESERVED: Reserved for future use. Value MUST be ignored.

ABF_ATTRIBUTE_FLAGS_EMAIL: If a bitwise AND of this value with the **Flags** field of the **ABF_ATTRIBUTE** structure yields a nonzero value, the associated attribute is an e-mail alias for an address book contact. If this bit is not set, the associated attribute does not represent an e-mail alias for an address book contact.

ABF_ATTRIBUTE_FLAGS_REQUIRED: If a bitwise AND of this value with the **Flags** field of the **ABF_ATTRIBUTE** structure yields a nonzero value, the associated attribute is required to be present on a user, contact or group object in order for the corresponding address book contact to be included in the address book file by the Address Book Server. If this bit is not set, the presence or absence of the associated attribute on a user, contact or group object does not affect whether the address book contact is included in the address book file. For information about how this attribute flag interacts with the other attribute flags, see section 7.

ABF_ATTRIBUTE_FLAGS_NC_MASK: This value can be used to mask out the phone number normalization control for an Active Directory attribute. Doing a bitwise AND of this value with the **Flags** field of the **ABF_ATTRIBUTE** yields a value from the **ABF_ATTRIBUTE_NORMALIZATION_CONTROL** enumeration.

ABF_ATTRIBUTE_FLAGS_EXCLUDE: If a bitwise AND of this value with the **Flags** field of the **ABF_ATTRIBUTE** yields a nonzero value, any user, contact or group object that has the associated attribute will be excluded from the address book file by the Address Book Server. This bit setting overrides the setting of the **ABF_ATTRIBUTE_FLAGS_INCLUDE** bit if both are specified for an attribute. If this bit is not set, the presence or absence of the associated attribute on a user, contact or group object does not affect whether the address book contact is

included in the address book file. For information about how this attribute flag interacts with the other attribute flags, see section 7.

ABF_ATTRIBUTE_FLAGS_INCLUDE: If a bitwise AND of this value with the **Flags** field of the **ABF_ATTRIBUTE** yields a nonzero value, then any user, contact or group object that has the associated attribute is included as an address book contact in the address book file by the Address Book Server. If this bit is not set, the presence or absence of the associated attribute on a user, contact or group object does not affect whether the address book contact is included in the address book file. For information about how this attribute flag interacts with the other attribute flags, see section 7.

ABF_ATTRIBUTE_FLAGS_GROUPTYPE: If a bitwise AND of this value with the **Flags** field of the **ABF_ATTRIBUTE** yields a nonzero value, the Boolean value of the associated attribute indicates whether the address book contact that has the associated attribute is a distribution list. If this bit is not set, the associated attribute is not used to determine if an address book contact is a distribution list.

ABF_ATTRIBUTE_FLAGS_DEVICE: If a bitwise AND of this value with the **Flags** field of the **ABF_ATTRIBUTE** yields a nonzero value, the associated attribute is included in the device-specific address book files (.dabs extension). If this bit is not set, the attribute is not included in the device-specific address book files. Group objects are never included in device specific address book files, so setting this flag on the groupType attribute has no effect.

ABF_ATTRIBUTE_FLAGS_UNUSED: These bits in the **Flags** field are unused and MUST be ignored.

ABF_ATTRIBUTE_FLAGS_CLIENT_MASK: Used to mask out the client mapping of an Active Directory attribute. Doing a bitwise AND of this value with the **Flags** field of the **ABF_ATTRIBUTE** yields a value from the **ABF_ATTRIBUTE_CLIENT_FIELD** enumeration.

2.1.2 ABF_ATTRIBUTE_TYPE Enumeration

The **ABF_ATTRIBUTE_TYPE** enumeration defines values that specify how the Address Book Server and clients process Active Directory attributes associated with address book contacts. These values are found in the low-order byte of the **Flags** field of the **ABF_ATTRIBUTE** structure.

```
typedef enum
{
  ABF_TYPE_STRING              = 0x00,
  ABF_TYPE_BINARY              = 0x01,
  ABF_TYPE_STRING_PROXYADDRESS = 0x02
} ABF_ATTRIBUTE_TYPE;
```

ABF_TYPE_STRING: An Active Directory attribute whose value is a Unicode string.

ABF_TYPE_BINARY: An Active Directory attribute whose value is a sequence of bytes.

ABF_TYPE_STRING_PROXYADDRESS: An Active Directory **proxyAddress**es attribute whose value is a Unicode string that MUST begin with "TEL:" or "SMTP:". The case of "tel:" and "smtp:" is not important. A **proxyAdddress**es attribute whose value starts with "SMTP:" MUST contain the '@' character in the value. A **proxyAdddress**es attribute whose value starts with "TEL:" MUST have the format "tel:+nnnnnnnnnnn;display-name=xxxxxxxxxx;ad-rdn=tttttt". **proxyAddress**es in Active Directory that begin with anything else (for example, "X500") are ignored and not stored in the Address Book Server output files. If **UseNormalizationRules** is set to "1", no **proxyAddress**es attribute values are included in the output files.

String "nnnnnnnnnnn" is up to 15 digits and represents the E164 form of the phone number, "xxxxxxxxxx" is arbitrary text to use as the display string for the phone number and "tttttt"

gives the type of phone number and may be **telephoneNumber**, **homeNumber**, **mobile**, or **otherTelephone**.

2.1.3 ABF_ATTRIBUTE_NORMALIZATION_CONTROL Enumeration

The **ABF_ATTRIBUTE_ NORMALIZATION_CONTROL** enumeration defines values that specify how an Address Book Server normalizes phone number attributes associated with user, contact and group objects. These values are found in bits 18 and 19 of the **Flags** field of the **ABF_ATTRIBUTE** structure(base index is 0). If normalization is enabled for a particular attribute, and that attribute is present multiple times for a given user or contact, only the first value of that attribute is normalized.

```
typedef enum
{
  ABF_NORMALIZATION_CONTROL_NONE         = 0x0000,
  ABF_NORMALIZATION_CONTROL_ALWAYS       = 0x1000,
  ABF_NORMALIZATION_CONTROL_PROXYADDRESS = 0x2000,
  ABF_NORMALIZATION_CONTROL_UNDEFINED    = 0x3000
} ABF_ATTRIBUTE_NORMALIZATION_CONTROL;
```

ABF_NORMALIZATION_CONTROL_NONE: Servers MUST NOT normalize the attribute value.

ABF_NORMALIZATION_CONTROL_ALWAYS: Servers MUST normalize the attribute value if matching the phone number normalization rules in **Rules** field of **ABF_NORMALIZATION_RULES** Structure; otherwise, the attribute value is not normalized.

ABF_NORMALIZATION_CONTROL_PROXYADDRESS: Servers MUST extract the normalized phone number from the **proxyAddress** value if the **UseNormalizationRules** field in **ABF_FULL_HEADER** or in **ABF_DELTA_HEADER** is zero. Otherwise, they MUST behave as if **ABF_NORMALIZATION_CONTROL_ALWAYS** is set.

Normalized numbers are stored in the **proxyAddress** value using the following syntax:

tel:+nnnnnnnnnnn;display-name=xxxxxxxxxx;ad-rdn=tttttt

where nnnnnnnnnnn is up to 15 digits and represents the E164 form of the phone number, xxxxxxxxxx is arbitrary text to use as the display string for the phone number and tttttt gives the type of phone number and may be telephoneNumber, homeNumber, mobile, or otherTelephone.

ABF_NORMALIZATION_CONTROL_UNDEFINED: This value is not used and MUST be ignored.

2.1.4 ABF_ATTRIBUTE_CLIENT_FIELD Enumeration

The **ABF_ATTRIBUTE_CLIENT_FIELD** enumeration defines values that specify how clients map attributes associated with address book contacts to attributes in the client database. These values are found in the high order byte of the **Flags** field of the **ABF_ATTRIBUTE** structure.

```
typedef enum
{
ABF_CLIENT_FIELD_PROXYADDRESSES = 0x00000000,
ABF_CLIENT_FIELD_GIVENNAME = 0x01000000,
ABF_CLIENT_FIELD_SN = 0x02000000,
ABF_CLIENT_FIELD_DISPLAYNAME = 0x03000000,
ABF_CLIENT_FIELD_TITLE = 0x04000000,
ABF_CLIENT_FIELD_MAILNICKNAME = 0x05000000,
ABF_CLIENT_FIELD_COMPANY = 0x06000000,
ABF_CLIENT_FIELD_PHYSICALDELIVERYOFFICENAME = 0x07000000,
ABF_CLIENT_FIELD_MSRTCSIP_PRIMARYUSERADDRESS = 0x08000000,
ABF_CLIENT_FIELD_TELEPHONENUMBER = 0x09000000,
ABF_CLIENT_FIELD_HOMENUMBER = 0x0A000000,
```

```
ABF_CLIENT_FIELD_MOBILE = 0x0B000000,
ABF_CLIENT_FIELD_OTHERTELEPHONE = 0x0C000000,
ABF_CLIENT_FIELD_IPPHONE = 0x0D000000,
ABF_CLIENT_FIELD_MAIL = 0x0E000000,
ABF_CLIENT_FIELD_GROUPTYPE = 0x0F000000,
ABF_CLIENT_FIELD_MANAGER = 0x10000000,
ABF_CLIENT_FIELD_IGNORE = 0xFF000000
} ABF_ATTRIBUTE_CLIENT_FIELD;
```

ABF_CLIENT_FIELD_PROXYADDRESSES: A **proxyAddress**es attribute of an address book contact.

ABF_CLIENT_FIELD_GIVENNAME: The first name of an address book contact.

ABF_CLIENT_FIELD_SN: The surname or last name of a user or contact.

ABF_CLIENT_FIELD_DISPLAYNAME: The display name of an address book contact.

ABF_CLIENT_FIELD_TITLE: The title of an address book contact.

ABF_CLIENT_FIELD_MAILNICKNAME: The e-mail account name of an address book contact.

ABF_CLIENT_FIELD_COMPANY: The company name of an address book contact.

ABF_CLIENT_FIELD_PHYSICALDELIVERYOFFICENAME: The office name of an address book contact.

ABF_CLIENT_FIELD_MSRTCSIP_PRIMARYUSERADDRESS: The primary SIP address of an address book contact.

ABF_CLIENT_FIELD_TELEPHONENUMBER: The work phone number of an address book contact.

ABF_CLIENT_FIELD_HOMENUMBER: The home phone number of an address book contact.

ABF_CLIENT_FIELD_MOBILE: The mobile phone number of an address book contact.

ABF_CLIENT_FIELD_OTHERTELEPHONE: An alternate phone number of an address book contact.

ABF_CLIENT_FIELD_IPPHONE: The IP phone number of an address book contact.

ABF_CLIENT_FIELD_MAIL: The email address (user@host) for an address book contact.

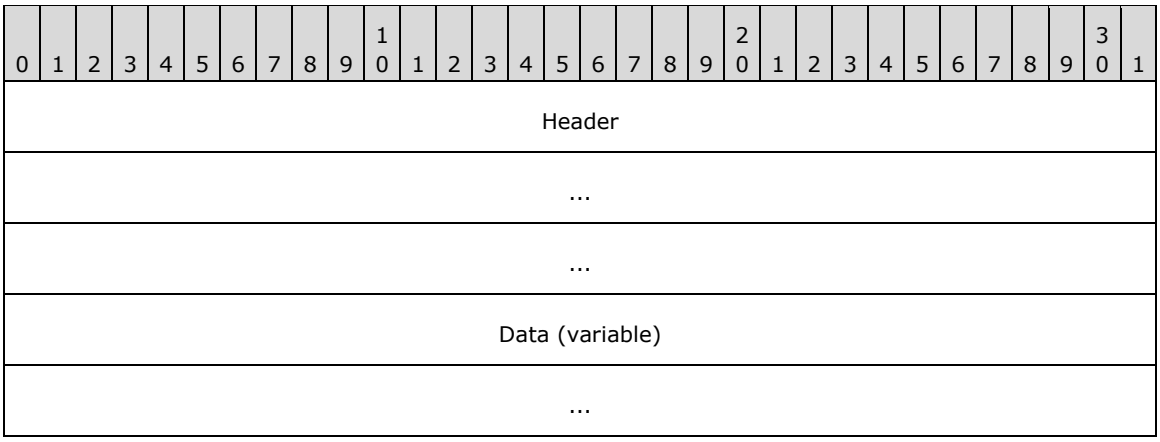
ABF_CLIENT_FIELD_GROUPTYPE: A Boolean indicator of whether an address book contact is a distribution list. If this attribute is present on an address book contact, the value MUST be 1, indicating that the address book contact is a distribution list.

ABF_CLIENT_FIELD_MANAGER: The **distinguished name (DN)** of the manager of an address book contact.<4>

ABF_CLIENT_FIELD_IGNORE: Ignored by the client and not mapped into any field in the client's database.

2.2 Address Book File Structures

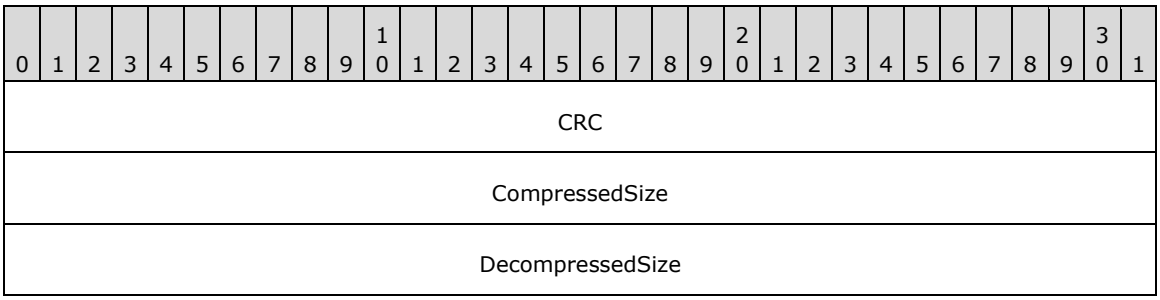
2.2.1 COMPRESSED_BLOCK Structure



Header (12 bytes): A **COMPRESSED_BLOCK_HEADER** structure that gives the **cyclic redundancy check (CRC)** for the original, uncompressed **Data** bytes, the length of the **Data** bytes, and the length of the decompressed **Data** bytes.

Data (variable): 1 or more bytes of compressed data. Length is defined in the **Header** structure.

2.2.2 COMPRESSED_BLOCK_HEADER Structure

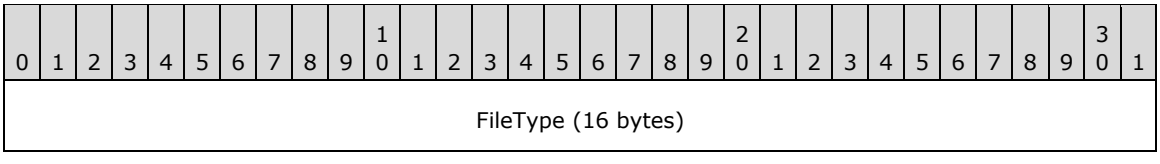


CRC (4 bytes): A 32-bit unsigned integer that gives the CRC for the original, uncompressed **Data** bytes. The value in this field MUST be calculated using the algorithm described in section 5.

CompressedSize (4 bytes): A 32-bit unsigned integer that gives the length of the **Data** bytes that follow the **Header** field. **CompressedSize** MUST be less than or equal to 64 kilobytes (0x10000).

DecompressedSize (4 bytes): A 32-bit unsigned integer that gives the length of the **Data** bytes after they are decompressed with the algorithm described in section 5. **DecompressedSize** MUST be greater than or equal to **CompressedSize**. If **CompressedSize** equals **DecompressedSize,** the block is not compressed, and the **Data** bytes represent the actual data. The block MAY be uncompressed.

2.2.3 ABF_FULL_HEADER Structure



...	
...	
NumberOfAttributes	CreationDate
UseNormalizationRules	MaximumAttributeId
SourceStream (128 bytes)	
...	
...	

FileType (16 bytes): A 16-byte **GUID** that MUST equal 0x76 0x6c 0xe1 0x44 0xfd 0x0a 0xa9 0x40 0x8b 0x63 0x5f 0xe9 0xb0 0x81 0x73 0x8f. This value indicates that the file is a full file.

CreationDate (2 bytes): A 16-bit unsigned number that is the XXXX portion of the F-XXXX file name.

NumberOfAttributes (2 bytes): A 16-bit unsigned number that is the number of **ABF_ATTRIBUTE** structures in the **ABF_ATTRIBUTES** structure.

MaximumAttribudeId (2 bytes): A 16-bit unsigned number that is the largest value of the **Id** field in all of the **ABF_ATTRIBUTE** structures in the **ABF_ATTRIBUTES** structure.<5>

UseNormalizationRules (2 bytes): A 16-bit unsigned number that is 0 if the server did not use the phone normalization rules in the **ABF_NORMALIZATION_RULES** structure to normalize phone number strings. The value is nonzero if the Address Book Server did normalize phone numbers using the phone normalization rules.

SourceStream (128 bytes): A 128-byte field that SHOULD be set to all zeroes. The client does not validate this field.

2.2.4 ABF_DELTA_HEADER Structure

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
FileType (16 bytes)																															
CreationDate																BaseCreationDate															
MaximumAttributeId																NumberOfAttributes															
SourceStream (128 bytes)																UseNormalizationRules															

FileType (16 bytes): A 16-byte GUID. For delta files, it MUST equal 0x16 0xc1 0x4b 0xb5 0x08 0x90 0xc7 0x47 0xb9 0xbd 0xf3 0xbb 0x1a 0x0a 0xb6 0xeb. This value indicates that the file is a delta file.

For compact files, it MUST equal 0x34 0x17 0x7d 0xf7 0x87 0xae 0x2b 0x4d 0x09 0xa0 0x8e 0xe9 0xba 0x89 0x4a 0x04. This value indicates that the file is a compact delta file.<6>

BaseCreationDate (2 bytes): A 16-bit unsigned number that is the XXXX portion of the D-XXXX-YYYY file name.

CreationDate (2 bytes): A 16-bit unsigned number that is the YYYY portion of the D-XXXX-YYYY file name.

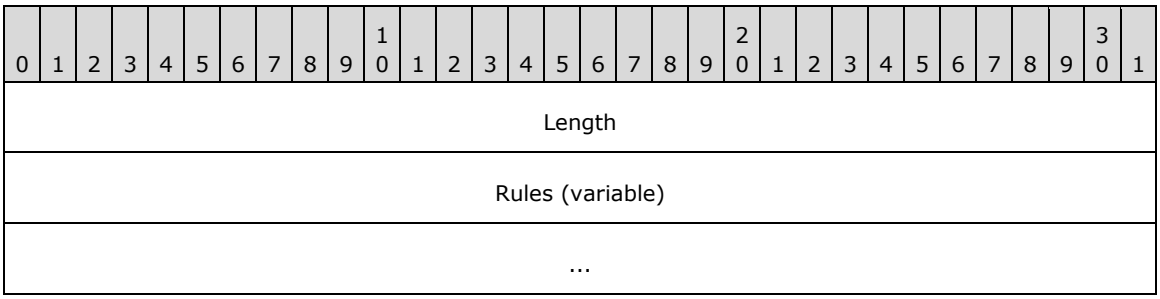
NumberOfAttributes (2 bytes): A 16-bit unsigned number that is the number of **ABF_ATTRIBUTE** structures in the **ABF_ATTRIBUTES** structure.

MaximumAttributeId (2 bytes): A 16-bit unsigned number that is the largest value of the **Id** field in all of the **ABF_ATTRIBUTE** structures in the **ABF_ATTRIBUTES** structure.<7>

UseNormalizationRules (2 bytes): A 16-bit unsigned number that is 0 if the server did not use the phone normalization rules in the **ABF_NORMALIZATION_RULES** structure to normalize phone number strings. The value is nonzero if the Address Book Server did normalize phone numbers using the phone normalization rules.

SourceStream (128 bytes): A 128-byte field that SHOULD be set to all zeroes. The client does not validate this field.

2.2.5 ABF_NORMALIZATION_RULES Structure

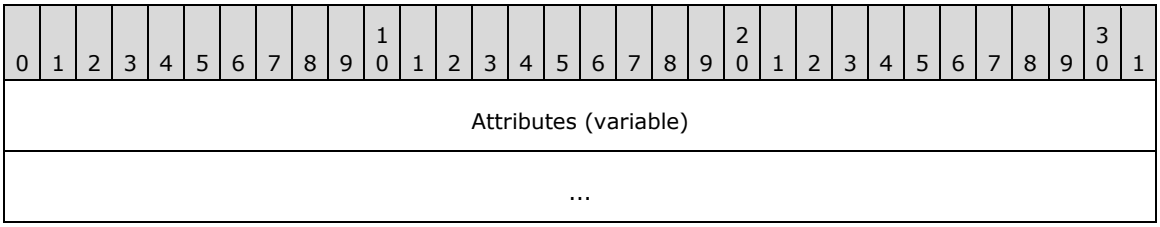


Length (4 bytes): A 32-bit unsigned number that is the number of bytes in the **Rules** field.

Rules (variable): A series of UTF-8 characters that contains phone number normalization rules. The number of characters in the series is specified by the **Length** field. If **Length** is nonzero, the last byte of the **Rules** field MUST be a zero byte. Each rule consists of two lines, and each line MUST be terminated by a carriage return, linefeed sequence (0xD, 0xA). The first line is a regular expression to match against a phone number string, using standard [MC-RegEx] regular expression syntax.<8> The second line is a replacement string to convert the matching phone number into a valid [E164] number. When more than one rule is present, the rules MUST be processed in order, and the phone number is normalized using the first matching rule. If no rule matches a phone number, the phone number is not normalized.

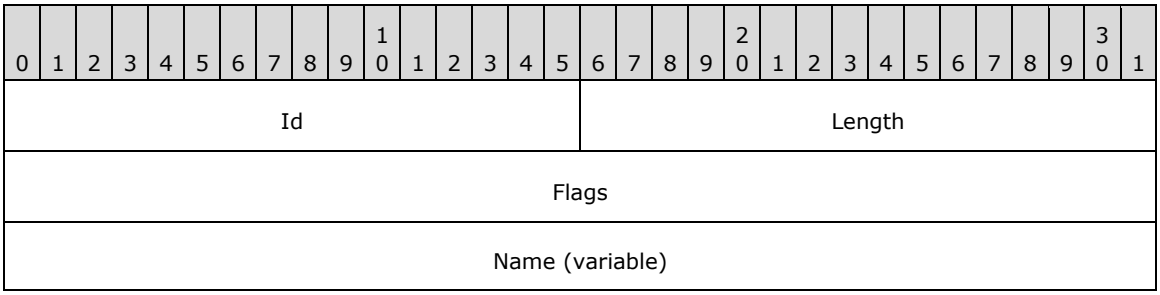
There is a special built-in rule called E164. If the regular expression in the first line of one of the preceding pairs of lines is the string "E164," the second line is ignored and the rule matches any input that consists of 15 or fewer decimal digits, with any spaces, periods, hyphens, or parentheses ignored. If an input does match the built-in E164 rule, the result is "tel:+" followed by the digits that matched. The ignored characters are discarded.

2.2.6 ABF_ATTRIBUTES Structure



Attributes (variable): One or more **ABF_ATTRIBUTE** structures.

2.2.7 ABF_ATTRIBUTE Structure



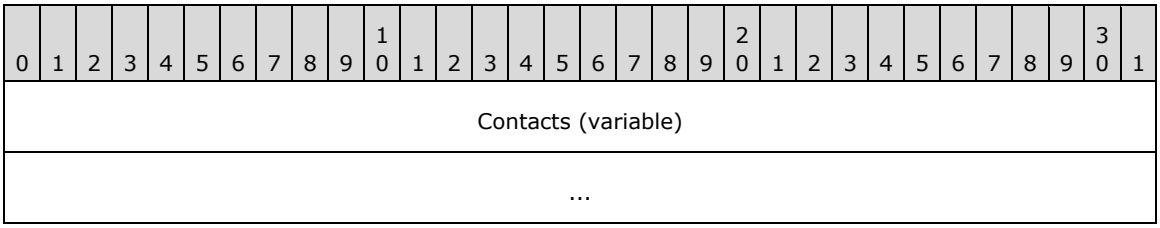
Length (2 bytes): A 16-bit unsigned integer that gives the length of the remaining bytes in the structure.

Id (2 bytes): A 16-bit unsigned integer that gives the identifier of this attribute as referenced by **ABF_CONTACT** structures and the **MaximumAttributeId** field of the **ABF_FULL_HEADER** and **ABF_DELTA_HEADER** structures.

Flags (4 bytes): A 32-bit unsigned integer that defines the type of attribute, how it is mapped by the client, and how it is processed by the Address Book Server. The bits in this field are described by the **ABF_ATTRIBUTE_FLAGS** enumeration.

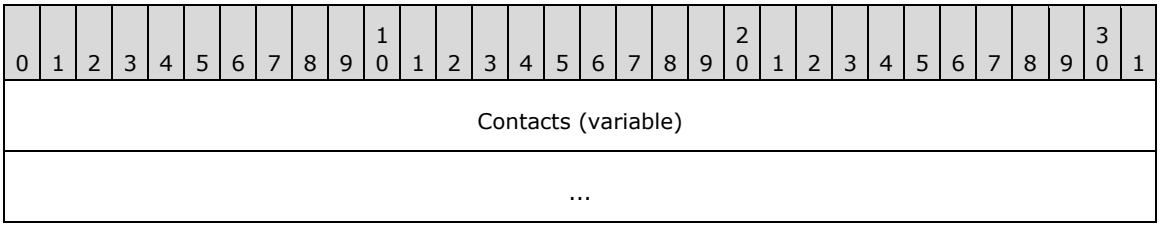
Name (variable): A zero-terminated UTF-8 string that is the name of the attribute in Active Directory. Name is not case-sensitive.

2.2.8 ABF_CONTACTS Structure



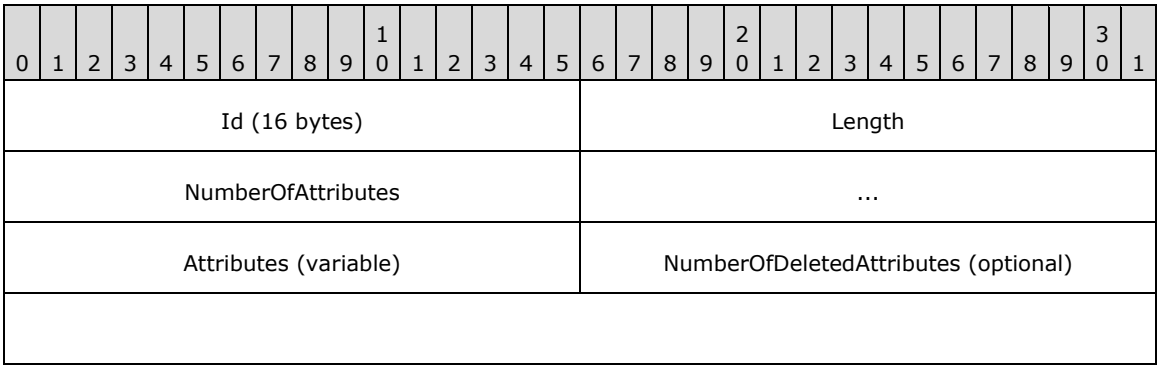
Contacts (variable): Zero or more **ABF_CONTACT** structures.

2.2.9 ABF_DELETED_CONTACTS Structure



Contacts (variable): Zero or more **ABF_CONTACT** structures.

2.2.10 ABF_CONTACT Structure



Length (2 bytes): A 16-bit unsigned integer that gives the length of the remaining bytes in the structure.

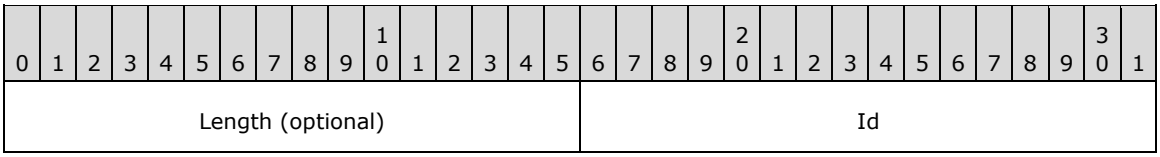
Id (16 bytes): A 16 byte GUID that is the value of the **objectGUID** Active Directory attribute for this address book contact. If this field is all zeroes, this is a sentinel entry that marks the end of the **ABF_CONTACTS** structure. The sentinel contact entry is not included in the count of **ABF_CONTACTS** structures. The value of the **NumberOfAttributes** field in the sentinel contact entry MUST be 0, and the **Length** field MUST be 0x12.

NumberOfAttributes (2 bytes): A 16-bit unsigned integer that gives the number of **ABF_CONTACT_ATTRIBUTE** records in the **Attributes** field.

NumberOfDeletedAttributes (2 bytes, optional): If the **NumberOfAttributes** field is zero, this is a deleted Contact object, and the **NumberOfDeletedAttributes** field MUST be present if the **Length** field is greater than 20 (0x14).

Attributes (variable): Zero or more **ABF_CONTACT_ATTRIBUTE** structures that give the requested Active Directory attributes for this address book contact object.

2.2.11 ABF_CONTACT_ATTRIBUTE Structure



Data (variable)

Id (1 or 2 bytes): An 8-bit or 16-bit unsigned integer that gives the identifier of the Active Directory attribute. This field corresponds to the **Id** field in the **ABF_ATTRIBUTE** structure. This field is 1 byte if the value of the **MaximumAttributeId** field in the **ABF_FULL_HEADER** or **ABF_DELTA_HEADER** structure is less than 256; otherwise, this field is 2 bytes.

Length (2 bytes, optional): A 16-bit unsigned integer that gives the length of the **Data** field if the **Id** field specifies a binary attribute. If the **Id** field does not specify a binary attribute, this field is not present.

Data (variable): A variable field. The preceding **Id** field is used to index the **ABF_ATTRIBUTES** array for the associated **ABF_ATTRIBUTE** structure. The **Flags** field of that structure, masked with the **ABF_ATTRIBUTE_FLAGS_TYPE_MASK,** specifies the type of this field. If the type is ABF_TYPE_BINARY, this field contains the bytes for that attribute. If the type is not ABF_TYPE_BINARY, this field contains a zero-terminated UTF-8 string.

2.2.12 ABF_FULL_TRAILER Structure

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
NumberOfContacts																Hash															
																NumberOfContacts (cont.)															

Hash (2 bytes): A 16-bit unsigned number that is a **hash** of the contents of the file. The **cryptographic hash function** used can be anything, but it MUST be the same function for all files. For an example of a hash function, see section 6.

NumberOfContacts (4 bytes): A 32-bit unsigned number that is the number of **ABF_CONTACT** structures in the file. Does not include the sentinel **ABF_CONTACT** structure (**Id** and **NumberOfAttributes** fields all zeroes).

2.2.13 ABF_DELTA_TRAILER Structure

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
BaseFileHash																Hash															
NumberOfContacts																															
NumberOfDeletedContacts																															

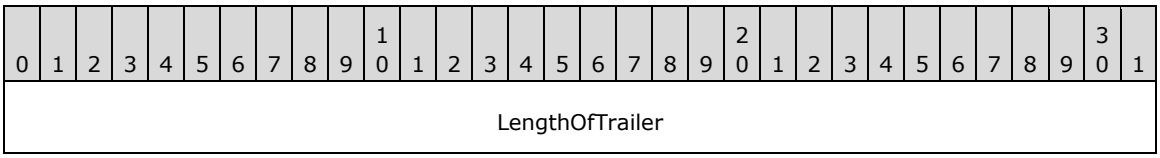
Hash (2 bytes): A 16-bit unsigned number that is a hash of the contents of the file. Any cryptographic hash function can be used, as long as it is the same function for all files. For an example of a hash function, see section 6.

BaseFileHash (2 bytes): A 16-bit unsigned number that is a hash of the contents of the base file that was used to calculate this delta file. Any hash function used can be used, but it MUST be the same function for all files. For an example of a hash function, see section 6.

NumberOfContacts (4 bytes): A 32-bit unsigned number that is the number of **ABF_CONTACT** structures in the file. Does not include the sentinel **ABF_CONTACT** structure (**Id** and **NumberOfAttributes** fields all zeroes).

NumberOfDeletedContacts (4 bytes): A 32-bit unsigned number that is the number of **ABF_CONTACT** structures in the file that identify deleted contacts.

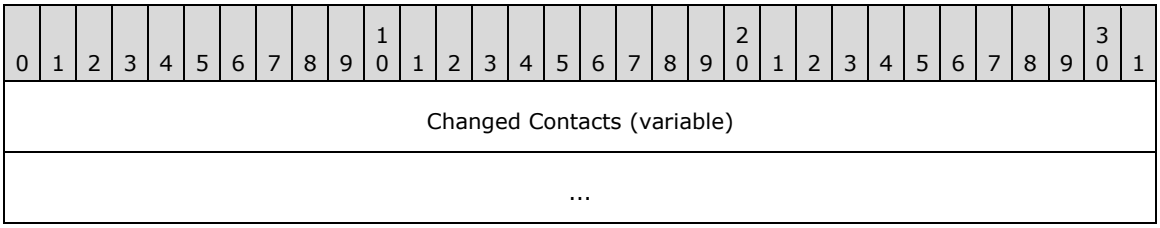
2.2.14 ABF_TRAILER_LENGTH Structure



LengthOfTrailer (4 bytes): A 32-bit unsigned number that is the length of either the **ABF_FULL_TRAILER** or **ABF_DELTA_TRAILER** structure that precedes this structure. Additional information MAY be added to the file format. If additional information is added, it MUST be placed after the **ABF_FULL_TRAILER** or **ABF_DELTA_TRAILER** structure and before the **ABF_TRAILER_LENGTH** structure. To access the trailer structure, readers of the file seek to the end of the file less the size of the **ABF_TRAILER_LENGTH** structure to read the length of the trailer structure. Once the **LengthOfTrailer** field has been read, readers of the file can seek from the end of the file less **LengthOfTrailer** less 4 (the size of the **ABF_TRAILER_LENGTH** structure) to get to the beginning of the trailer structure. This allows future versions of the file format to contain more information if necessary. If more information is added to the file, the **LengthOfTrailer** MUST include the length of the added information. Using this mechanism prevents breaking old clients using the existing file format.

2.2.15 ABF_CONTACTS_CHANGES Structure

The product behavior in this section replaces that in section 2.2.8.<9>



Changed Contacts (variable): Zero or more **ABF_CONTACT_CHANGES** structures. Compact delta files uses ABF_CONTACTS_CHANGES structure instead of ABF_CONTACTS structure.

2.2.16 ABF_CONTACT_CHANGES Structure

The product behavior in this section replaces that in section 2.2.10.<10>

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
Id (16 bytes)																Length															
NumberOfAttributes																...															
Attributes (variable)																NumberOfDeletedAttributes (optional)															

Length (2 bytes): A 16-bit unsigned integer that gives the length of the remaining bytes in the structure.

Id (16 bytes): A 16 byte GUID that is the value of the **objectGUID** Active Directory attribute for this **address book contact**. If this field is all zeroes, this is a sentinel entry that marks the end of the **ABF_CONTACTS_CHANGES** structure. The sentinel contact entry is not included in the count of **ABF_CONTACTS_CHANGES** structures. The value of the **NumberOfAttributes** field in the sentinel contact entry MUST be 0, and the **Length** field MUST be 0x12.

NumberOfAttributes (2 bytes): A 16-bit unsigned integer that gives the number of **ABF_CONTACT_ATTRIBUTE** records in the **Attributes** field.

NumberOfDeletedAttributes (2 bytes, optional): If the **NumberOfAttributes** field is zero, this is a deleted Contact object, and the **NumberOfDeletedAttributes** field MUST be present if the **Length** field is greater than 20 (0x14).

Attributes (variable): Zero or more **ABF_CONTACT_ATTRIBUTE** structures that give the requested Active Directory attributes for this **address book contact** object. These are only the attributes that have changed for the contact. If an attribute is deleted, it shows up here as empty. Missing attributes are unchanged.

3 Structure Examples

3.1 Address Book File

The primary function of the Address Book Server is to scan Active Directory for users, contacts and groups, and determine which objects to include in the Address Book Server files and, for each object included, which attributes to include. For pseudo-code that shows how users, contacts, and groups are chosen from Active Directory, see section 7.

This section describes an example of an address book output file. The scenario that produced this address book file is as follows:

- Monday, January 21, 2008, a **full address book file** is produced, F-0A10.lsabs.
- Later in the day, an administrator makes several edits to the Active Directory Domain Services. The **displayName** attribute of one user is changed, another user is deleted, and a new user is added.
- Tuesday, January 22, 2008, a full address book file is produced, F-0A11.lsabs, along with a **delta address book file**, D-0A10-0A11.lsabs, which is the delta between the two full files. and a compact delta address book file, C-0A10-0A11.lsabs, which is the delta of the contact attribute changes between the two full files.<11>
- The compact delta file is identical to the delta file with two exceptions. 1) only the attributes that changed for a contact are included, so unchanged attributes are ignored, and 2) attributes for the contact that no longer exist are marked as empty. <12>

The file that will be used in the example is the delta address book file, D-0A10-0A11.lsabs.

The relevant portion of Active Directory contents when F-0A10.lsabs is generated:

```
        sn: ABSUser1 lastname;
        title: Development Manager;
        physicalDeliveryOfficeName: 12345;
        telephoneNumber: 555 391 3224;
        givenName: ABSUser1 firstname;
        displayName: ABSUser1_displayname;
        otherTelephone: 555-533-4312;
        company: TestCompany;
        proxyAddresses: sip:ABSUser1@urtest.rtmp.selfhost.corp.proseware.com;
        otherHomePhone: 555-391-3042;
        objectGUID: 8c36ad0a-5e97-46dd-8d5b-255140c52b00;
        mail: ABSUser1@urtest.com;
        homePhone: 555-566-4312;
        mobile: 555-533-4313;
        msRTCSIP-PrimaryUserAddress: sip:ABSUser1@urtest.rtmp.selfhost.corp.proseware.com;
    -----------

    Dn: CN=ABSUser2,CN=Users,...
        sn: ABSUser2_lastname;
        telephoneNumber: 555-783-4756;
        givenName: ABSUser2_firstname;
        displayName: ABSUser2 displayname;
        proxyAddresses: sip:ABSUser2@urtest.rtmp.selfhost.corp.proseware.com;
        objectGUID: 477c251b-db42-4ef6-9c69-fabbecc67f31;
        msRTCSIP-PrimaryUserAddress: sip:ABSUser2@urtest.rtmp.selfhost.corp.proseware.com;
    -----------

    Dn: CN=ABSUser3,CN=Users,...
        sn: ABSUser3_lastname;
        title: Program Manager;
        telephoneNumber: 555-555-1234;
        givenName: ABSUser3_firstname;
        displayName: ABSUser3_displayname;
```

```
    company: TestCompany;
    proxyAddresses: sip:ABSUser3@urtest.rtmp.selfhost.corp.proseware.com;
    otherHomePhone: 555-555-5678;
    objectGUID: e335ddce-8a89-4e38-a4eb-116965911f4a;
    mail: ABSUser3@urtest.com;
    homePhone: 555-555-4321;
    msRTCSIP-PrimaryUserAddress: sip:ABSUser3@urtest.rtmp.selfhost.corp.proseware.com;
-----------

Dn: CN=ABSUser4,CN=Users,...
    sn: ABSUser4 lastname;
    title: Program Manager;
    telephoneNumber: 555-555-8765;
    givenName: ABSUser4_firstname;
    displayName: ABSUser4_displayname;
    company: TestCompany;
    proxyAddresses: sip:ABSUser4@urtest.rtmp.selfhost.corp.proseware.com;
    objectGUID: f02291c7-4c80-4956-8708-6bd526f47be5;
    mail: ABSUser4@urtest.com;
    mobile: 555-555-1111;
    msRTCSIP-PrimaryUserAddress: sip:ABSUser4@urtest.rtmp.selfhost.corp.proseware.com;
-----------
```

The relevant portion of the Active Directory contents when F-0A11.lsabs is generated:

```
Dn: CN=ABSUser1,CN=Users,...
    sn: ABSUser1_lastname;
    title: Development Manager;
    physicalDeliveryOfficeName: 12345;
    telephoneNumber: 555 391 3224;
    givenName: ABSUser1 firstname;
    displayName: ABSUser1_displayname_changed;
    otherTelephone: 555-533-4312;
    company: TestCompany;
    proxyAddresses: sip:ABSUser1@urtest.rtmp.selfhost.corp.proseware.com;
    otherHomePhone: 555-391-3042;
    objectGUID: 8c36ad0a-5e97-46dd-8d5b-255140c52b00;
    mail: ABSUser1@urtest.com;
    homePhone: 555-566-4312;
    mobile: 555-533-4313;
    msRTCSIP-PrimaryUserAddress: sip:ABSUser1@urtest.rtmp.selfhost.corp.proseware.com;
-----------

Dn: CN=ABSUser3,CN=Users,...
    sn: ABSUser3_lastname;
    title: Program Manager;
    telephoneNumber: 555-555-1234;
    givenName: ABSUser3_firstname;
    displayName: ABSUser3 displayname;
    company: TestCompany;
    proxyAddresses: sip:ABSUser3@urtest.rtmp.selfhost.corp.proseware.com;
    otherHomePhone: 555-555-5678;
    objectGUID: e335ddce-8a89-4e38-a4eb-116965911f4a;
    mail: ABSUser3@urtest.com;
    homePhone: 555-555-4321;
    msRTCSIP-PrimaryUserAddress: sip:ABSUser3@urtest.rtmp.selfhost.corp.proseware.com;
-----------

Dn: CN=ABSUser4,CN=Users,...
    sn: ABSUser4 lastname;
    title: Program Manager;
    telephoneNumber: 555-555-8765;
    givenName: ABSUser4_firstname;
    displayName: ABSUser4_displayname;
    company: TestCompany;
    proxyAddresses: sip:ABSUser4@urtest.rtmp.selfhost.corp.proseware.com;
    objectGUID: f02291c7-4c80-4956-8708-6bd526f47be5;
```

```
    mail: ABSUser4@urtest.com;
    mobile: 555-555-1111;
    msRTCSIP-PrimaryUserAddress: sip:ABSUser4@urtest.rtmp.selfhost.corp.proseware.com;
-----------

Dn: CN=ABSUser5,CN=Users,...
    sn: ABSUser5_lastname;
    telephoneNumber: 555-789-6666;
    givenName: ABSUser5_firstname;
    displayName: ABSUser5_displayname;
    proxyAddresses: sip:ABSUser5@urtest.rtmp.selfhost.corp.proseware.com;
    objectGUID: e1e8410b-c8c4-4022-93ba-b2145ed6d134;
    msRTCSIP-PrimaryUserAddress: sip:ABSUser5@urtest.rtmp.selfhost.corp.proseware.com;
-----------
```

The simple text difference of the two preceding Active Directory listings follows, to illustrate what was changed. The **displayName** attribute for ABSUser1 was changed, ABSUser2 was deleted, and a new user, ABSUser5, was added.

```
7c7
<     displayName: ABSUser1_displayname;
---
>     displayName: ABSUser1_displayname_changed;
19,28d18
< Dn: CN=ABSUser2,CN=Users,...
<     sn: ABSUser2_lastname;
<     telephoneNumber: 555-783-4756;
<     givenName: ABSUser2_firstname;
<     displayName: ABSUser2_displayname;
<     proxyAddresses: sip:ABSUser2@urtest.rtmp.selfhost.corp.proseware.com;
<     objectGUID: 477c251b-db42-4ef6-9c69-fabbecc67f31;
<     msRTCSIP-PrimaryUserAddress: sip:ABSUser2@urtest.rtmp.selfhost.corp.proseware.com;
< -----------
<
56a47,56
>
> Dn: CN=ABSUser5,CN=Users,...
>     sn: ABSUser5_lastname;
>     telephoneNumber: 555-789-6666;
>     givenName: ABSUser5_firstname;
>     displayName: ABSUser5_displayname;
>     proxyAddresses: sip:ABSUser5@urtest.rtmp.selfhost.corp.proseware.com;
>     objectGUID: e1e8410b-c8c4-4022-93ba-b2145ed6d134;
>     msRTCSIP-PrimaryUserAddress: sip:ABSUser5@urtest.rtmp.selfhost.corp.proseware.com;
> -----------
```

The contents of this delta address book file example follow, in hexadecimal bytes. The far-left column is the byte count; the far-right characters are the interpretation of the bytes in the **American National Standards Institute (ANSI) character set**. The sections that follow describe the structures that convey this series of bytes. This hexadecimal dump occurred after the actual D-0A10-0A11.lsabs file was decompressed. For a description of how the file is decompressed, see section 5.

```
00000000: 16 c1 4b b5 08 90 c7 47 b9 bd f3 bb 1a 0a b6 eb ..K....G........
00000010: 10 0a 11 0a 14 00 14 00 01 00 00 00 00 00 00 00 ................
00000020: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
00000030: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
00000040: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
00000050: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
00000060: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
00000070: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
00000080: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
00000090: 00 00 00 00 00 00 00 00 00 00 5b 08 00 00 2e 2a ..........[....*
000000a0: 38 38 32 5b 5c 73 28 29 5c 2d 5c 2e 2f 5d 2a 38 882[\s()\-\./]*8
```

```
000000b0: 30 38 30 2e 2a 5b 58 78 5d 2b 5b 5c 73 28 29 5c 080.*[Xx]+[\s()\
000000c0: 2d 5c 2e 2f 5d 2a 28 5c 64 5c 64 5c 64 5c 64 5c -\./]*(\d\d\d\d\
000000d0: 64 29 0d 0a 24 31 3b 70 68 6f 6e 65 2d 63 6f 6e d)..$1;phone-con
000000e0: 74 65 78 74 3d 70 72 6F 73 65 77 61 72 65 2e 63 text=proseware.c
000000f0: 6f 6d 0d 0a 2e 2a 38 38 32 5b 5c 73 28 29 5c 2d om...*882[\s()\-
00000100: 5c 2e 2f 5d 2a 38 30 38 30 2e 2a 0d 0a 6e 75 6c \./]*8080.*..nul
00000110: 6c 0d 0a 5c 28 28 5c 2b 5c 73 2a 31 29 3f 5c 29 l..\((\+\s*1)?\)
00000120: 5b 5c 73 5c 2d 5c 2e 5d 2a 5c 28 3f 28 5c 64 5c [\s\-\.]*\(?(\d\
00000130: 64 5c 64 29 5c 73 2a 5c 29 3f 5b 5c 73 28 29 5c d\d)\s*\)?[\s()\
00000140: 2d 5c 2e 2f 5d 2a 28 5c 64 5c 64 5c 64 29 5b 5c -\./]*(\d\d\d)[\
00000150: 73 28 29 5c 2d 5c 2e 2f 5d 2a 28 5c 64 5c 64 5c s()\-\./]*(\d\d\
00000160: 64 5c 64 29 5c 73 2a 5b 58 78 5d 2b 5b 5c 73 28 d\d)\s*[Xx]+[\s(
00000170: 29 5c 2d 5c 2e 2f 5d 2a 28 5c 64 5c 64 5c 64 5c )\-\./]*(\d\d\d\
00000180: 64 5c 64 29 5b 5c 73 28 29 5c 2d 5c 2e 2f 5d 2a d\d)[\s()\-\./]*
00000190: 0d 0a 2b 31 24 32 24 33 24 34 3b 65 78 74 3d 24 ..+1$2$3$4;ext=$
000001a0: 35 0d 0a 28 5c 2b 5c 73 2a 31 29 3f 5b 5c 73 5c 5..(\+\s*1)?[\s\
000001b0: 2d 5c 2e 5d 2a 5c 28 3f 28 5c 64 5c 64 5c 64 29 -\.]*\(?(\d\d\d)
000001c0: 5c 73 2a 5c 29 3f 5b 5c 73 28 29 5c 2d 5c 2e 2f \s*\)?[\s()\-\./
000001d0: 5d 2a 28 5c 64 5c 64 5c 64 29 5b 5c 73 28 29 5c ]*(\d\d\d)[\s()\
000001e0: 2d 5c 2e 2f 5d 2a 28 5c 64 5c 64 5c 64 5c 64 29 -\./]*(\d\d\d\d)
000001f0: 5c 73 2a 45 58 54 5c 73 2a 28 5c 64 5c 64 5c 64 \s*EXT\s*(\d\d\d
00000200: 5c 64 5c 64 29 0d 0a 2b 31 24 32 24 33 24 34 3b \d\d)..+1$2$3$4;
00000210: 65 78 74 3d 24 35 0d 0a 28 5c 2b 5c 73 2a 31 29 ext=$5..(\+\s*1)
00000220: 3f 5b 5c 73 5c 2d 5c 2e 5d 2a 5c 28 3f 28 5c 64 ?[\s\-\.]*\(?(\d
00000230: 5c 64 5c 64 29 5c 73 2a 5c 29 3f 5b 5c 73 28 29 \d\d)\s*\)?[\s()
00000240: 5c 2d 5c 2e 2f 5d 2a 28 5c 64 5c 64 5c 64 29 5b \-\./]*(\d\d\d)[
00000250: 5c 73 28 29 5c 2d 5c 2e 2f 5d 2a 28 5c 64 5c 64 \s()\-\./]*(\d\d
00000260: 5c 64 5c 64 29 5c 73 2a 5b 78 58 5d 5c 73 2a 28 \d\d)\s*[xX]\s*(
00000270: 5c 64 5c 64 5c 64 5c 64 5c 64 29 0d 0a 2b 31 24 \d\d\d\d\d)..+1$
00000280: 32 24 33 24 34 3b 65 78 74 3d 24 35 0d 0a 28 5c 2$3$4;ext=$5..(\
00000290: 73 2a 31 29 5b 5c 73 5c 2d 5c 2e 5d 2a 5c 28 3f s*1)[\s\-\.]*\(?
000002a0: 5c 73 2a 28 5c 64 5c 64 5c 64 29 5c 73 2a 5c 29 \s*(\d\d\d)\s*\)
000002b0: 3f 5b 5c 73 28 29 5c 2d 5c 2e 2f 5d 2a 28 5c 64 ?[\s()\-\./]*(\d
000002c0: 5c 64 5c 64 29 5b 5c 73 28 29 5c 2d 5c 2e 2f 5d \d\d)[\s()\-\./]
000002d0: 2a 28 5c 64 5c 64 5c 64 5c 64 29 5b 5c 73 5d 2a *(\d\d\d\d)[\s]*
000002e0: 0d 0a 2b 31 24 32 24 33 24 34 0d 0a 28 5c 2b 5c ..+1$2$3$4..(\+\
000002f0: 73 2a 31 29 3f 5b 5c 73 5c 2d 5c 2e 5d 2a 5c 28 s*1)?[\s\-\.]*\(
00000300: 3f 28 5c 64 5c 64 5c 64 29 5c 73 2a 5c 29 3f 5b ?(\d\d\d)\s*\)?[
00000310: 5c 73 28 29 5c 2d 5c 2e 2f 5d 2a 37 30 28 5c 64 \s()\-\./]*70(\d
00000320: 5c 64 5c 64 5c 64 5c 64 29 0d 0a 2b 31 24 32 37 \d\d\d\d)..+1$27
00000330: 30 24 33 3b 65 78 74 3d 24 33 0d 0a 37 30 28 5c 0$3;ext=$3..70(\
00000340: 64 29 5b 5c 73 28 29 5c 2d 5c 2e 2f 5d 2a 28 5c d)[\s()\-\./]*(\
00000350: 64 5c 64 5c 64 5c 64 29 5c 73 2a 5b 58 78 5d 2b d\d\d\d)\s*[Xx]+
00000360: 28 5c 64 5c 64 5c 64 5c 64 5c 64 29 0d 0a 2b 31 (\d\d\d\d\d)..+1
00000370: 34 32 35 37 30 24 31 24 32 3b 65 78 74 3d 24 33 42570$1$2;ext=$3
00000380: 0d 0a 28 5c 64 5c 64 5c 64 29 5b 5c 73 28 29 5c ..(\d\d\d)[\s()\
00000390: 2d 5c 2e 2f 5d 2a 28 5c 64 5c 64 5c 64 5c 64 29 -\./]*(\d\d\d\d)
000003a0: 5c 73 2a 5b 58 78 5d 2b 28 5c 64 5c 64 5c 64 5c \s*[Xx]+(\d\d\d\
000003b0: 64 5c 64 29 0d 0a 2b 31 34 32 35 24 31 24 32 3b d\d)..+1425$1$2;
000003c0: 65 78 74 3d 24 33 0d 0a 28 5c 64 5c 64 5c 64 29 ext=$3..(\d\d\d)
000003d0: 5b 5c 73 28 29 5c 2d 5c 2e 2f 5d 2a 28 5c 64 5c [\s()\-\./]*(\d\
000003e0: 64 5c 64 5c 64 29 0d 0a 2b 31 34 32 35 24 31 24 d\d\d)..+1425$1$
000003f0: 32 0d 0a 5b 58 78 5d 5c 73 2a 28 5c 64 5c 64 5c 2..[Xx]\s*(\d\d\
00000400: 64 5c 64 5c 64 29 0d 0a 24 31 3b 70 68 6f 6e 65 d\d\d)..$1;phone
00000410: 2d 63 6f 6e 74 65 78 74 3d 70 72 6F 73 65 77 61 -context=prosewa
00000420: 72 65 2e 63 6f 6d 0d 0a 5c 28 3f 28 5c 64 5c 64 re.com..\(?(\d\d
00000430: 29 5c 73 2a 5c 29 3f 5c 73 2a 5b 58 78 5d 2b 5c )\s*\)?\s*[Xx]+\
00000440: 73 2a 28 5c 64 5c 64 5c 64 5c 64 5c 64 29 0d 0a s*(\d\d\d\d\d)..
00000450: 2b 31 34 32 35 24 31 24 32 3b 65 78 74 3d 24 32 +1425$1$2;ext=$2
00000460: 0d 0a 30 31 31 28 5c 64 2b 29 28 5b 5c 73 28 29 ..011(\d+)([\s()
00000470: 5c 2d 5c 2e 2f 5d 2b 28 5c 64 2b 29 29 3f 28 5b \-\./]+(\d+))?([
00000480: 5c 73 28 29 5c 2d 5c 2e 2f 5d 2b 28 5c 64 2b 29 \s()\-\./]+(\d+)
00000490: 29 3f 28 5b 5c 73 28 29 5c 2d 5c 2e 2f 5d 2b 28 )?([\s()\-\./]+(
000004a0: 5c 64 2b 29 29 3f 28 5b 5c 73 28 29 5c 2d 5c 2e \d+))?([\s()\-\.
000004b0: 2f 5d 2b 28 5c 64 2b 29 29 3f 28 5b 5c 73 28 29 /]+(\d+))?([\s()
000004c0: 5c 2d 5c 2e 2f 5d 2b 28 5c 64 2b 29 29 3f 28 5b \-\./]+(\d+))?([
000004d0: 5c 73 28 29 5c 2d 5c 2e 2f 5d 2b 28 5c 64 2b 29 \s()\-\./]+(\d+)
000004e0: 29 3f 28 5b 5c 73 28 29 5c 2d 5c 2e 2f 5d 2b 28 )?([\s()\-\./]+(
000004f0: 5c 64 2b 29 29 3f 28 5b 5c 73 28 29 5c 2d 5c 2e \d+))?([\s()\-\.
```

```
00000500: 2f 5d 2b 28 5c 64 2b 29 29 3f 28 5b 5c 73 28 29 /]+(\d+))?([\s()
00000510: 5c 2d 5c 2e 2f 5d 2b 28 5c 64 2b 29 29 3f 28 5b \-\./]+(\d+))?([
00000520: 5c 73 28 29 5c 2d 5c 2e 2f 5d 2b 28 5c 64 2b 29 \s()\-\./]+(\d+)
00000530: 29 3f 5c 73 2a 5b 58 78 5d 2b 28 5c 64 7b 31 2c )?\s*[Xx]+(\d{1,
00000540: 31 35 7d 29 5b 5c 73 5d 2a 0d 0a 2b 24 31 24 33 15})[\s]*..+$1$3
00000550: 24 35 24 37 24 39 24 31 31 24 31 33 24 31 35 24 $5$7$9$11$13$15$
00000560: 31 37 24 31 39 24 32 31 3b 65 78 74 3d 24 32 32 17$19$21;ext=$22
00000570: 0d 0a 30 31 31 28 5c 64 2b 29 28 5b 5c 73 28 29 ..011(\d+)([\s()
00000580: 5c 2d 5c 2e 2f 5d 2b 28 5c 64 2b 29 29 3f 28 5b \-\./]+(\d+))?([
00000590: 5c 73 28 29 5c 2d 5c 2e 2f 5d 2b 28 5c 64 2b 29 \s()\-\./]+(\d+)
000005a0: 29 3f 28 5b 5c 73 28 29 5c 2d 5c 2e 2f 5d 2b 28 )?([\s()\-\./]+(
000005b0: 5c 64 2b 29 29 3f 28 5b 5c 73 28 29 5c 2d 5c 2e \d+))?([\s()\-\.
000005c0: 2f 5d 2b 28 5c 64 2b 29 29 3f 28 5b 5c 73 28 29 /]+(\d+))?([\s()
000005d0: 5c 2d 5c 2e 2f 5d 2b 28 5c 64 2b 29 29 3f 28 5b \-\./]+(\d+))?([
000005e0: 5c 73 28 29 5c 2d 5c 2e 2f 5d 2b 28 5c 64 2b 29 \s()\-\./]+(\d+)
000005f0: 29 3f 28 5b 5c 73 28 29 5c 2d 5c 2e 2f 5d 2b 28 )?([\s()\-\./]+(
00000600: 5c 64 2b 29 29 3f 28 5b 5c 73 28 29 5c 2d 5c 2e \d+))?([\s()\-\.
00000610: 2f 5d 2b 28 5c 64 2b 29 29 3f 28 5b 5c 73 28 29 /]+(\d+))?([\s()
00000620: 5c 2d 5c 2e 2f 5d 2b 28 5c 64 2b 29 29 3f 28 5b \-\./]+(\d+))?([
00000630: 5c 73 28 29 5c 2d 5c 2e 2f 5d 2b 28 5c 64 2b 29 \s()\-\./]+(\d+)
00000640: 29 3f 5b 5c 73 5d 2a 0d 0a 2b 24 31 24 33 24 35 )?[\s]*..+$1$3$5
00000650: 24 37 24 39 24 31 31 24 31 33 24 31 35 24 31 37 $7$9$11$13$15$17
00000660: 24 31 39 24 32 31 0d 0a 28 5c 64 2b 29 28 5b 5c $19$21..(\d+)([\
00000670: 73 28 29 5c 2d 5c 2e 2f 5d 2b 28 5c 64 2b 29 29 s()\-\./]+(\d+))
00000680: 28 5b 5c 73 28 29 5c 2d 5c 2e 2f 5d 2b 28 5c 64 ([\s()\-\./]+(\d
00000690: 2b 29 29 28 5b 5c 73 28 29 5c 2d 5c 2e 2f 5d 2b +))([\s()\-\./]+
000006a0: 28 5c 64 2b 29 29 3f 28 5b 5c 73 28 29 5c 2d 5c (\d+))?([\s()\-\
000006b0: 2e 2f 5d 2b 28 5c 64 2b 29 29 3f 28 5b 5c 73 28 ./]+(\d+))?([\s(
000006c0: 29 5c 2d 5c 2e 2f 5d 2b 28 5c 64 2b 29 29 3f 5b )\-\./]+(\d+))?[
000006d0: 5c 73 5d 2a 0d 0a 2b 24 31 24 33 24 35 24 37 24 \s]*..+$1$3$5$7$
000006e0: 39 24 31 31 0d 0a 45 31 36 34 0d 0a 6e 75 6c 6c 9$11..E164..null
000006f0: 0d 0a 5c 2b 2b 28 5c 64 2b 29 28 5b 5c 73 28 29 ..\++(\d+)([\s()
00000700: 5c 2d 5c 2e 2f 5d 2b 28 5c 64 2b 29 29 3f 28 5b \-\./]+(\d+))?([
00000710: 5c 73 28 29 5c 2d 5c 2e 2f 5d 2b 28 5c 64 2b 29 \s()\-\./]+(\d+)
00000720: 29 3f 28 5b 5c 73 28 29 5c 2d 5c 2e 2f 5d 2b 28 )?([\s()\-\./]+(
00000730: 5c 64 2b 29 29 3f 28 5b 5c 73 28 29 5c 2d 5c 2e \d+))?([\s()\-\.
00000740: 2f 5d 2b 28 5c 64 2b 29 29 3f 28 5b 5c 73 28 29 /]+(\d+))?([\s()
00000750: 5c 2d 5c 2e 2f 5d 2b 28 5c 64 2b 29 29 3f 28 5b \-\./]+(\d+))?([
00000760: 5c 73 28 29 5c 2d 5c 2e 2f 5d 2b 28 5c 64 2b 29 \s()\-\./]+(\d+)
00000770: 29 3f 28 5b 5c 73 28 29 5c 2d 5c 2e 2f 5d 2b 28 )?([\s()\-\./]+(
00000780: 5c 64 2b 29 29 3f 28 5b 5c 73 28 29 5c 2d 5c 2e \d+))?([\s()\-\.
00000790: 2f 5d 2b 28 5c 64 2b 29 29 3f 28 5b 5c 73 28 29 /]+(\d+))?([\s()
000007a0: 5c 2d 5c 2e 2f 5d 2b 28 5c 64 2b 29 29 3f 28 5b \-\./]+(\d+))?([
000007b0: 5c 73 28 29 5c 2d 5c 2e 2f 5d 2b 28 5c 64 2b 29 \s()\-\./]+(\d+)
000007c0: 29 3f 5c 73 2a 5b 58 78 5d 2b 28 5c 64 7b 31 2c )?\s*[Xx]+(\d{1,
000007d0: 31 35 7d 29 5b 5c 73 5d 2a 0d 0a 2b 24 31 24 33 15})[\s]*..+$1$3
000007e0: 24 35 24 37 24 39 24 31 31 24 31 33 24 31 35 24 $5$7$9$11$13$15$
000007f0: 31 37 24 31 39 24 32 31 3b 65 78 74 3d 24 32 32 17$19$21;ext=$22
00000800: 0d 0a 5c 2b 2b 28 5c 64 2b 29 28 5b 5c 73 28 29 ..\++(\d+)([\s()
00000810: 5c 2d 5c 2e 2f 5d 2b 28 5c 64 2b 29 29 3f 28 5b \-\./]+(\d+))?([
00000820: 5c 73 28 29 5c 2d 5c 2e 2f 5d 2b 28 5c 64 2b 29 \s()\-\./]+(\d+)
00000830: 29 3f 28 5b 5c 73 28 29 5c 2d 5c 2e 2f 5d 2b 28 )?([\s()\-\./]+(
00000840: 5c 64 2b 29 29 3f 28 5b 5c 73 28 29 5c 2d 5c 2e \d+))?([\s()\-\.
00000850: 2f 5d 2b 28 5c 64 2b 29 29 3f 28 5b 5c 73 28 29 /]+(\d+))?([\s()
00000860: 5c 2d 5c 2e 2f 5d 2b 28 5c 64 2b 29 29 3f 28 5b \-\./]+(\d+))?([
00000870: 5c 73 28 29 5c 2d 5c 2e 2f 5d 2b 28 5c 64 2b 29 \s()\-\./]+(\d+)
00000880: 29 3f 28 5b 5c 73 28 29 5c 2d 5c 2e 2f 5d 2b 28 )?([\s()\-\./]+(
00000890: 5c 64 2b 29 29 3f 28 5b 5c 73 28 29 5c 2d 5c 2e \d+))?([\s()\-\.
000008a0: 2f 5d 2b 28 5c 64 2b 29 29 3f 28 5b 5c 73 28 29 /]+(\d+))?([\s()
000008b0: 5c 2d 5c 2e 2f 5d 2b 28 5c 64 2b 29 29 3f 28 5b \-\./]+(\d+))?([
000008c0: 5c 73 28 29 5c 2d 5c 2e 2f 5d 2b 28 5c 64 2b 29 \s()\-\./]+(\d+)
000008d0: 29 3f 5b 5c 73 5d 2a 0d 0a 2b 24 31 24 33 24 35 )?[\s]*..+$1$3$5
000008e0: 24 37 24 39 24 31 31 24 31 33 24 31 35 24 31 37 $7$9$11$13$15$17
000008f0: 24 31 39 24 32 31 0d 0a 00 0e 00 14 00 00 00 00 $19$21........
00000900: 10 6d 61 6e 61 67 65 72 00 10 00 13 00 01 08 01 .manager........
00000910: 0f 67 72 6f 75 70 54 79 70 65 00 15 00 12 00 02 .groupType......
00000920: 01 00 00 70 72 6f 78 79 41 64 64 72 65 73 73 65 ...proxyAddresse
00000930: 73 00 0b 00 11 00 00 00 00 0e 6d 61 69 6c 00 0e s.........mail..
00000940: 00 10 00 00 20 00 0d 69 70 50 68 6f 6e 65 00 15 .... ..ipPhone..
```

```
00000950: 00 0f 00 00 20 00 0c 6f 74 68 65 72 54 65 6c 65 .... ..otherTele
00000960: 70 68 6f 6e 65 00 12 00 0e 00 00 20 00 0b 6f 74 phone...... ..ot
00000970: 68 65 72 4d 6f 62 69 6c 65 00 0d 00 0d 00 00 28 herMobile......(
00000980: 02 0b 6d 6f 62 69 6c 65 00 15 00 0c 00 00 20 00 ..mobile...... .
00000990: 0a 6f 74 68 65 72 48 6f 6d 65 50 68 6f 6e 65 00 .otherHomePhone.
000009a0: 10 00 0b 00 00 28 00 0a 68 6f 6d 65 50 68 6f 6e .....(..homePhon
000009b0: 65 00 16 00 0a 00 00 28 02 09 74 65 6c 65 70 68 e......(..teleph
000009c0: 6f 6e 65 4e 75 6d 62 65 72 00 22 00 09 00 00 08 oneNumber."..... 
000009d0: 02 08 6d 73 52 54 43 53 49 50 2d 50 72 69 6d 61 ..msRTCSIP-Prima
000009e0: 72 79 55 73 65 72 41 64 64 72 65 73 73 00 21 00 ryUserAddress.!.
000009f0: 08 00 00 00 00 07 70 68 79 73 69 63 61 6c 44 65 ......physicalDe
00000a00: 6c 69 76 65 72 79 4f 66 66 69 63 65 4e 61 6d 65 liveryOfficeName
00000a10: 00 0e 00 07 00 00 00 00 06 63 6f 6d 70 61 6e 79 .........company
00000a20: 00 13 00 06 00 00 04 00 05 6d 61 69 6c 4e 69 63 .........mailNic
00000a30: 6b 6e 61 6d 65 00 0c 00 05 00 00 00 00 04 74 69 kname.........ti
00000a40: 74 6c 65 00 12 00 04 00 00 00 02 03 64 69 73 70 tle.........disp
00000a50: 6c 61 79 4e 61 6d 65 00 09 00 03 00 00 00 00 02 layName.........
00000a60: 73 6e 00 10 00 02 00 00 00 00 01 67 69 76 65 6e sn.........given
00000a70: 4e 61 6d 65 00 21 00 01 00 00 00 00 ff 6d 73 45 Name.!.......msE
00000a80: 78 63 68 48 69 64 65 46 72 6f 6d 41 64 64 72 65 xchHideFromAddre
00000a90: 73 73 4c 69 73 74 73 00 66 01 0a ad 36 8c 97 5e ssLists.f...6..^
00000aa0: dd 46 8d 5b 25 51 40 c5 2b 00 12 00 11 41 42 53 .F.[%Q@.+....ABS
00000ab0: 55 73 65 72 31 40 75 72 74 65 73 74 2e 63 6f 6d User1@urtest.com
00000ac0: 00 0f 35 35 35 2d 35 33 33 2d 34 33 31 32 00 0f ..555-533-4312..
00000ad0: 74 65 6c 3a 2b 35 35 35 35 33 33 34 33 31 32 00 tel:+5555334312.
00000ae0: 0d 35 35 35 2d 35 33 33 2d 34 33 31 33 00 0d 74 .555-533-4313..t
00000af0: 65 6c 3a 2b 35 35 35 35 33 33 34 33 31 33 00 0c el:+5555334313..
00000b00: 35 35 35 2d 33 39 31 2d 33 30 34 32 00 0c 74 65 555-391-3042..te
00000b10: 6c 3a 2b 35 35 35 33 39 31 33 30 34 32 00 0b 35 l:+5553913042..5
00000b20: 35 35 2d 35 36 36 2d 34 33 31 32 00 0b 74 65 6c 55-566-4312..tel
00000b30: 3a 2b 35 35 35 35 36 36 34 33 31 32 00 0a 35 35 :+5555664312..55
00000b40: 35 20 33 39 31 20 33 32 32 34 00 0a 74 65 6c 3a 5 391 3224..tel:
00000b50: 2b 35 35 35 33 39 31 33 32 32 34 00 09 73 69 70 +5553913224..sip
00000b60: 3a 41 42 53 55 73 65 72 31 40 75 72 74 65 73 74 :ABSUser1@urtest
00000b70: 2e 72 74 6d 70 2e 73 65 6c 66 68 6f 73 74 2e 63 .rtmp.selfhost.c
00000b80: 6f 72 70 2e 70 72 6F 73 65 77 61 72 65 2e 63 6f orp.proseware.co
00000b90: 6d 00 08 31 32 33 34 35 00 07 54 65 73 74 43 6f m..12345..TestCo
00000ba0: 6d 70 61 6e 79 00 05 44 65 76 65 6c 6f 70 6d 65 mpany..Developme
00000bb0: 6e 74 20 4d 61 6e 61 67 65 72 00 04 41 42 53 55 nt Manager..ABSU
00000bc0: 73 65 72 31 5f 64 69 73 70 6c 61 79 6e 61 6d 65 ser1 displayname
00000bd0: 5f 63 68 61 6e 67 65 64 00 03 41 42 53 55 73 65 _changed..ABSUse
00000be0: 72 31 5f 6c 61 73 74 6e 61 6d 65 00 02 41 42 53 r1 lastname..ABS
00000bf0: 55 73 65 72 31 5f 66 69 72 73 74 6e 61 6d 65 00 User1_firstname.
00000c00: a4 00 0b 41 e8 e1 c4 c8 22 40 93 ba b2 14 5e d6 ...A...."@....^.
00000c10: d1 34 06 00 0a 35 35 35 2d 37 38 39 2d 36 36 36 .4...555-789-666
00000c20: 36 00 0a 74 65 6c 3a 2b 35 35 35 37 38 39 36 36 6..tel:+55578966
00000c30: 36 36 00 09 73 69 70 3a 41 42 53 55 73 65 72 35 66..sip:ABSUser5
00000c40: 40 75 72 74 65 73 74 2e 72 74 6d 70 2e 73 65 6c @urtest.rtmp.sel
00000c50: 66 68 6f 73 74 2e 63 6f 72 70 2e 70 72 6F 73 65 fhost.corp.prose
00000c60: 77 61 72 65 2e 63 6f 6d 00 04 41 42 53 55 73 65 ware.com..ABSUse
00000c70: 72 35 5f 64 69 73 70 6c 61 79 6e 61 6d 65 00 03 r5_displayname..
00000c80: 41 42 53 55 73 65 72 35 5f 6c 61 73 74 6e 61 6d ABSUser5 lastnam
00000c90: 65 00 02 41 42 53 55 73 65 72 35 5f 66 69 72 73 e..ABSUser5_firs
00000ca0: 74 6e 61 6d 65 00 a6 00 1b 25 7c 47 42 db f6 4e tname....%|GB..N
00000cb0: 9c 69 fa bb ec c6 7f 31 00 00 06 00 0a 35 35 35 .i.....1.....555
00000cc0: 2d 37 38 33 2d 34 37 35 36 00 0a 74 65 6c 3a 2b -783-4756..tel:+
00000cd0: 35 35 35 37 38 33 34 37 35 36 00 09 73 69 70 3a 5557834756..sip:
00000ce0: 41 42 53 55 73 65 72 32 40 75 72 74 65 73 74 2e ABSUser2@urtest.
00000cf0: 72 74 6d 70 2e 73 65 6c 66 68 6f 73 74 2e 63 6f rtmp.selfhost.co
00000d00: 72 70 2e 70 72 6F 73 65 77 61 72 65 2e 63 6f 6d rp.proseware.com
00000d10: 00 04 41 42 53 55 73 65 72 32 5f 64 69 73 70 6c ..ABSUser2_displ
00000d20: 61 79 6e 61 6d 65 00 03 41 42 53 55 73 65 72 32 ayname..ABSUser2
00000d30: 5f 6c 61 73 74 6e 61 6d 65 00 02 41 42 53 55 73  lastname..ABSUs
00000d40: 65 72 32 5f 66 69 72 73 74 6e 61 6d 65 00 12 00 er2_firstname...
00000d50: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
00000d60: 00 00 db df d2 dd 02 00 00 00 01 00 00 00 0c 00 ................
00000d70: 00 00                                          ..
```

3.1.1 ABF_DELTA_HEADER

This section provides an example of the address book file **ABF_DELTA_HEADER** structure specified in section 2.2.4.

```
00000000: 16 C1 4B B5 08 90 C7 47 B9 BD F3 BB 1A 0A B6 EB ..K....G........
00000010: 10 0A 11 0A 14 00 14 00 01 00 00 00 00 00 00 00 ................
00000020: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
00000030: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
00000040: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
00000050: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
00000060: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
00000070: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
00000080: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
00000090: 00 00 00 00 00 00 00 00 00 00                   ..........
```

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
FileType (0xB54BC116)																															
0x47C79008																															
0xBBF3BDB9																															
0xEBB60A1A																															
CreationDate (0x0A11)																BaseCreationDate (0x0A10)															
MaximumAttributeId (0x0014)																NumberOfAttributes (0x0014)															
SourceStream (0x0000)																UseNormalizationRules (0x0001)															
126 more bytes of 0x00																															

FileType: The GUID value {b54bc116-9008-47c7-b9bd-f3bb1a0ab6eb} indicates that this is a delta address book file.

BaseCreationDate: 0x0A10 specifies the base full file (F-0A10.lsabs) that was used to create this delta file.

CreationDate: 0x0A11 specifies the date that this delta file (D-0A10-0A11.lsabs) was created, as well as the full file (F-0A11.lsabs) that was compared against the base full file (F-0A10.lsabs) to produce this delta file.

NumberOfAttributes: 0x0014 specifies the number of **ABF_ATTRIBUTE** structures in the **ABF_ATTRIBUTES** structure.

MaximumAttributeId: 0x0014 specifies the largest value of the **Id** field in all of the **ABF_ATTRIBUTE** structures in the **ABF_ATTRIBUTES** structure.

UseNormalizationRules: 0x0001 indicates that the Address Book Server did normalize phone numbers using the phone normalization rules in the **ABF_NORMALIZATION_RULES** structure.

SourceStream: 128 bytes of 0x00 are ignored by the client.

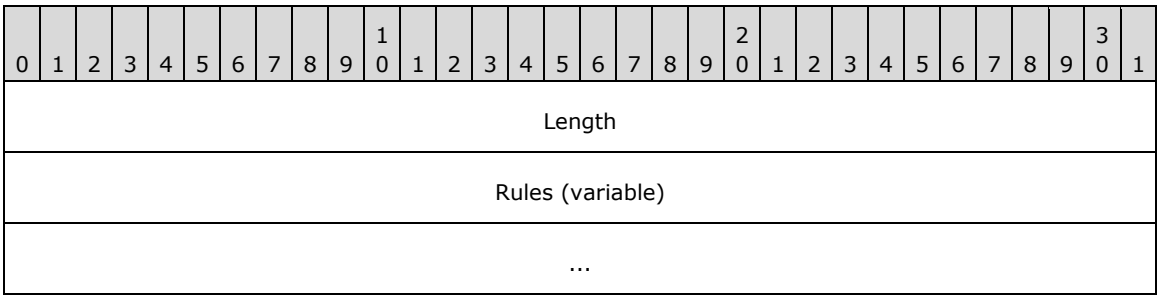
3.1.2 ABF_NORMALIZATION_RULES

This section provides an example of the address book file **ABF_NORMALIZATION_RULES** structure specified in section 2.2.5.

```
00000090:                                 5B 08 00 00 2E 2A           [....*
000000A0: 38 38 32 5B 5C 73 28 29 5C 2D 5C 2E 2F 5D 2A 38 882[\s()\-\./]*8
000000B0: 30 38 30 2E 2A 5B 58 78 5D 2B 5B 5C 73 28 29 5C 080.*[Xx]+[\s()\
000000C0: 2D 5C 2E 2F 5D 2A 28 5C 64 5C 64 5C 64 5C 64 5C -\./]*(\d\d\d\d\
000000D0: 64 29 0D 0A 24 31 3B 70 68 6F 6E 65 2D 63 6F 6E d)..$1;phone-con
000000E0: 74 65 78 74 3D 70 72 6F 73 65 77 61 72 65 2E 63 text=proseware.c
000000F0: 6F 6D 0D 0A 2E 2A 38 38 32 5B 5C 73 28 29 5C 2D om...*882[\s()\-
00000100: 5C 2E 2F 5D 2A 38 30 38 30 2E 2A 0D 0A 6E 75 6C \./]*8080.*..nul
00000110: 6C 0D 0A 5C 28 28 5C 2B 5C 73 2A 31 29 3F 5C 29 l..\((\+\s*1)?\)
00000120: 5B 5C 73 5C 2D 5C 2E 5D 2A 5C 28 3F 28 5C 64 5C [\s\-\.]*\(?(\d\
00000130: 64 5C 64 29 5C 73 2A 5C 29 3F 5B 5C 73 28 29 5C d\d)\s*\)?[\s()\
00000140: 2D 5C 2E 2F 5D 2A 28 5C 64 5C 64 5C 64 29 5B 5C -\./]*(\d\d\d)[\
00000150: 73 28 29 5C 2D 5C 2E 2F 5D 2A 28 5C 64 5C 64 5C s()\-\./]*(\d\d\
00000160: 64 5C 64 29 5C 73 2A 5B 58 78 5D 2B 5B 5C 73 28 d\d)\s*[Xx]+[\s(
00000170: 29 5C 2D 5C 2E 2F 5D 2A 28 5C 64 5C 64 5C 64 5C )\-\./]*(\d\d\d\
00000180: 64 5C 64 29 5B 5C 73 28 29 5C 2D 5C 2E 2F 5D 2A d\d)[\s()\-\./]*
00000190: 0D 0A 2B 31 24 32 24 33 24 34 3B 65 78 74 3D 24 ..+1$2$3$4;ext=$
000001A0: 35 0D 0A 28 5C 2B 5C 73 2A 31 29 3F 5B 5C 73 5C 5..(\+\s*1)?[\s\
000001B0: 2D 5C 2E 5D 2A 5C 28 3F 28 5C 64 5C 64 5C 64 29 -\.]*\(?(\d\d\d)
000001C0: 5C 73 2A 5C 29 3F 5B 5C 73 28 29 5C 2D 5C 2E 2F \s*\)?[\s()\-\./
000001D0: 5D 2A 28 5C 64 5C 64 5C 64 29 5B 5C 73 28 29 5C ]*(\d\d\d)[\s()\
000001E0: 2D 5C 2E 2F 5D 2A 28 5C 64 5C 64 5C 64 5C 64 29 -\./]*(\d\d\d\d)
000001F0: 5C 73 2A 45 58 54 5C 73 2A 28 5C 64 5C 64 5C 64 \s*EXT\s*(\d\d\d
00000200: 5C 64 5C 64 29 0D 0A 2B 31 24 32 24 33 24 34 3B \d\d)..+1$2$3$4;
00000210: 65 78 74 3D 24 35 0D 0A 28 5C 2B 5C 73 2A 31 29 ext=$5..(\+\s*1)
00000220: 3F 5B 5C 73 5C 2D 5C 2E 5D 2A 5C 28 3F 28 5C 64 ?[\s\-\.]*\(?(\d
00000230: 5C 64 5C 64 29 5C 73 2A 5C 29 3F 5B 5C 73 28 29 \d\d)\s*\)?[\s()
00000240: 5C 2D 5C 2E 2F 5D 2A 28 5C 64 5C 64 5C 64 29 5B \-\./]*(\d\d\d)[
00000250: 5C 73 28 29 5C 2D 5C 2E 2F 5D 2A 28 5C 64 5C 64 \s()\-\./]*(\d\d
00000260: 5C 64 5C 64 29 5C 73 2A 5B 78 58 5D 5C 73 2A 28 \d\d)\s*[xX]\s*(
00000270: 5C 64 5C 64 5C 64 5C 64 5C 64 29 0D 0A 2B 31 24 \d\d\d\d\d)..+1$
00000280: 32 24 33 24 34 3B 65 78 74 3D 24 35 0D 0A 28 5C 2$3$4;ext=$5..(\
00000290: 73 2A 31 29 5B 5C 73 5C 2D 5C 2E 5D 2A 5C 28 3F s*1)[\s\-\.]*\(?
000002A0: 5C 73 2A 28 5C 64 5C 64 5C 64 29 5C 73 2A 5C 29 \s*(\d\d\d)\s*\)
000002B0: 3F 5B 5C 73 28 29 5C 2D 5C 2E 2F 5D 2A 28 5C 64 ?[\s()\-\./]*(\d
000002C0: 5C 64 5C 64 29 5B 5C 73 28 29 5C 2D 5C 2E 2F 5D \d\d)[\s()\-\./]
000002D0: 2A 28 5C 64 5C 64 5C 64 5C 64 29 5B 5C 73 5D 2A *(\d\d\d\d)[\s]*
000002E0: 0D 0A 2B 31 24 32 24 33 24 34 0D 0A 28 5C 2B 5C ..+1$2$3$4..(\+\
000002F0: 73 2A 31 29 3F 5B 5C 73 5C 2D 5C 2E 5D 2A 5C 28 s*1)?[\s\-\.]*\(
00000300: 3F 28 5C 64 5C 64 5C 64 29 5C 73 2A 5C 29 3F 5B ?(\d\d\d)\s*\)?[
00000310: 5C 73 28 29 5C 2D 5C 2E 2F 5D 2A 37 30 28 5C 64 \s()\-\./]*70(\d
00000320: 5C 64 5C 64 5C 64 5C 64 29 0D 0A 2B 31 24 32 37 \d\d\d\d)..+1$27
00000330: 30 24 33 3B 65 78 74 3D 24 33 0D 0A 37 30 28 5C 0$3;ext=$3..70(\
00000340: 64 29 5B 5C 73 28 29 5C 2D 5C 2E 2F 5D 2A 28 5C d)[\s()\-\./]*(\
00000350: 64 5C 64 5C 64 5C 64 29 5C 73 2A 5B 58 78 5D 2B d\d\d\d)\s*[Xx]+
00000360: 28 5C 64 5C 64 5C 64 5C 64 5C 64 29 0D 0A 2B 31 (\d\d\d\d\d)..+1
00000370: 34 32 35 37 30 24 31 24 32 3B 65 78 74 3D 24 33 42570$1$2;ext=$3
00000380: 0D 0A 28 5C 64 5C 64 5C 64 29 5B 5C 73 28 29 5C ..(\d\d\d)[\s()\
00000390: 2D 5C 2E 2F 5D 2A 28 5C 64 5C 64 5C 64 5C 64 29 -\./]*(\d\d\d\d)
000003A0: 5C 73 2A 5B 58 78 5D 2B 28 5C 64 5C 64 5C 64 5C \s*[Xx]+(\d\d\d\
000003B0: 64 5C 64 29 0D 0A 2B 31 34 32 35 24 31 24 32 3B d\d)..+1425$1$2;
000003C0: 65 78 74 3D 24 33 0D 0A 28 5C 64 5C 64 5C 64 29 ext=$3..(\d\d\d)
000003D0: 5B 5C 73 28 29 5C 2D 5C 2E 2F 5D 2A 28 5C 64 5C [\s()\-\./]*(\d\
000003E0: 64 5C 64 5C 64 29 0D 0A 2B 31 34 32 35 24 31 24 d\d\d)..+1425$1$
000003F0: 32 0D 0A 5B 58 78 5D 5C 73 2A 28 5C 64 5C 64 5C 2..[Xx]\s*(\d\d\
00000400: 64 5C 64 5C 64 29 0D 0A 24 31 3B 70 68 6F 6E 65 d\d\d)..$1;phone
00000410: 2D 63 6F 6E 74 65 78 74 3D 70 72 6F 73 65 77 61 -context=prosewa
00000420: 72 65 2E 63 6F 6D 0D 0A 5C 28 3F 28 5C 64 5C 64 re.com..\(?(\d\d
00000430: 29 5C 73 2A 5C 29 3F 5C 73 2A 5B 58 78 5D 2B 5C )\s*\)?\s*[Xx]+\
00000440: 73 2A 28 5C 64 5C 64 5C 64 5C 64 5C 64 29 0D 0A s*(\d\d\d\d\d)..
```

```
00000450: 2B 31 34 32 35 24 31 24 32 3B 65 78 74 3D 24 32 +1425$1$2;ext=$2
00000460: 0D 0A 30 31 31 28 5C 64 2B 29 28 5B 5C 73 28 29 ..011(\d+)([\s()
00000470: 5C 2D 5C 2E 2F 5D 2B 28 5C 64 2B 29 29 3F 28 5B \-\./]+(\d+))?([
00000480: 5C 73 28 29 5C 2D 5C 2E 2F 5D 2B 28 5C 64 2B 29 \s()\-\./]+(\d+)
00000490: 29 3F 28 5B 5C 73 28 29 5C 2D 5C 2E 2F 5D 2B 28 )?([\s()\-\./]+(
000004A0: 5C 64 2B 29 29 3F 28 5B 5C 73 28 29 5C 2D 5C 2E \d+))?([\s()\-\.
000004B0: 2F 5D 2B 28 5C 64 2B 29 29 3F 28 5B 5C 73 28 29 /]+(\d+))?([\s()
000004C0: 5C 2D 5C 2E 2F 5D 2B 28 5C 64 2B 29 29 3F 28 5B \-\./]+(\d+))?([
000004D0: 5C 73 28 29 5C 2D 5C 2E 2F 5D 2B 28 5C 64 2B 29 \s()\-\./]+(\d+)
000004E0: 29 3F 28 5B 5C 73 28 29 5C 2D 5C 2E 2F 5D 2B 28 )?([\s()\-\./]+(
000004F0: 5C 64 2B 29 29 3F 28 5B 5C 73 28 29 5C 2D 5C 2E \d+))?([\s()\-\.
00000500: 2F 5D 2B 28 5C 64 2B 29 29 3F 28 5B 5C 73 28 29 /]+(\d+))?([\s()
00000510: 5C 2D 5C 2E 2F 5D 2B 28 5C 64 2B 29 29 3F 28 5B \-\./]+(\d+))?([
00000520: 5C 73 28 29 5C 2D 5C 2E 2F 5D 2B 28 5C 64 2B 29 \s()\-\./]+(\d+)
00000530: 29 3F 5C 73 2A 5B 58 78 5D 2B 28 5C 64 7B 31 2C )?\s*[Xx]+(\d{1,
00000540: 31 35 7D 29 5B 5C 73 5D 2A 0D 0A 2B 24 31 24 33 15})[\s]*..+$1$3
00000550: 24 35 24 37 24 39 24 31 31 24 31 33 24 31 35 24 $5$7$9$11$13$15$
00000560: 31 37 24 31 39 24 32 31 3B 65 78 74 3D 24 32 32 17$19$21;ext=$22
00000570: 0D 0A 30 31 31 28 5C 64 2B 29 28 5B 5C 73 28 29 ..011(\d+)([\s()
00000580: 5C 2D 5C 2E 2F 5D 2B 28 5C 64 2B 29 29 3F 28 5B \-\./]+(\d+))?([
00000590: 5C 73 28 29 5C 2D 5C 2E 2F 5D 2B 28 5C 64 2B 29 \s()\-\./]+(\d+)
000005A0: 29 3F 28 5B 5C 73 28 29 5C 2D 5C 2E 2F 5D 2B 28 )?([\s()\-\./]+(
000005B0: 5C 64 2B 29 29 3F 28 5B 5C 73 28 29 5C 2D 5C 2E \d+))?([\s()\-\.
000005C0: 2F 5D 2B 28 5C 64 2B 29 29 3F 28 5B 5C 73 28 29 /]+(\d+))?([\s()
000005D0: 5C 2D 5C 2E 2F 5D 2B 28 5C 64 2B 29 29 3F 28 5B \-\./]+(\d+))?([
000005E0: 5C 73 28 29 5C 2D 5C 2E 2F 5D 2B 28 5C 64 2B 29 \s()\-\./]+(\d+)
000005F0: 29 3F 28 5B 5C 73 28 29 5C 2D 5C 2E 2F 5D 2B 28 )?([\s()\-\./]+(
00000600: 5C 64 2B 29 29 3F 28 5B 5C 73 28 29 5C 2D 5C 2E \d+))?([\s()\-\.
00000610: 2F 5D 2B 28 5C 64 2B 29 29 3F 28 5B 5C 73 28 29 /]+(\d+))?([\s()
00000620: 5C 2D 5C 2E 2F 5D 2B 28 5C 64 2B 29 29 3F 28 5B \-\./]+(\d+))?([
00000630: 5C 73 28 29 5C 2D 5C 2E 2F 5D 2B 28 5C 64 2B 29 \s()\-\./]+(\d+)
00000640: 29 3F 5B 5C 73 5D 2A 0D 0A 2B 24 31 24 33 24 35 )?[\s]*..+$1$3$5
00000650: 24 37 24 39 24 31 31 24 31 33 24 31 35 24 31 37 $7$9$11$13$15$17
00000660: 24 31 39 24 32 31 0D 0A 28 5C 64 2B 29 28 5B 5C $19$21..(\d+)([\
00000670: 73 28 29 5C 2D 5C 2E 2F 5D 2B 28 5C 64 2B 29 29 s()\-\./]+(\d+))
00000680: 28 5B 5C 73 28 29 5C 2D 5C 2E 2F 5D 2B 28 5C 64 ([\s()\-\./]+(\d
00000690: 2B 29 29 28 5B 5C 73 28 29 5C 2D 5C 2E 2F 5D 2B +))([\s()\-\./]+
000006A0: 28 5C 64 2B 29 29 3F 28 5B 5C 73 28 29 5C 2D 5C (\d+))?([\s()\-\
000006B0: 2E 2F 5D 2B 28 5C 64 2B 29 29 3F 28 5B 5C 73 28 ./]+(\d+))?([\s(
000006C0: 29 5C 2D 5C 2E 2F 5D 2B 28 5C 64 2B 29 29 3F 5B )\-\./]+(\d+))?[
000006D0: 5C 73 5D 2A 0D 0A 2B 24 31 24 33 24 35 24 37 24 \s]*..+$1$3$5$7$
000006E0: 39 24 31 31 0D 0A 45 31 36 34 0D 0A 6E 75 6C 6C 9$11..E164..null
000006F0: 0D 0A 5C 2B 2B 28 5C 64 2B 29 28 5B 5C 73 28 29 ..\++(\d+)([\s()
00000700: 5C 2D 5C 2E 2F 5D 2B 28 5C 64 2B 29 29 3F 28 5B \-\./]+(\d+))?([
00000710: 5C 73 28 29 5C 2D 5C 2E 2F 5D 2B 28 5C 64 2B 29 \s()\-\./]+(\d+)
00000720: 29 3F 28 5B 5C 73 28 29 5C 2D 5C 2E 2F 5D 2B 28 )?([\s()\-\./]+(
00000730: 5C 64 2B 29 29 3F 28 5B 5C 73 28 29 5C 2D 5C 2E \d+))?([\s()\-\.
00000740: 2F 5D 2B 28 5C 64 2B 29 29 3F 28 5B 5C 73 28 29 /]+(\d+))?([\s()
00000750: 5C 2D 5C 2E 2F 5D 2B 28 5C 64 2B 29 29 3F 28 5B \-\./]+(\d+))?([
00000760: 5C 73 28 29 5C 2D 5C 2E 2F 5D 2B 28 5C 64 2B 29 \s()\-\./]+(\d+)
00000770: 29 3F 28 5B 5C 73 28 29 5C 2D 5C 2E 2F 5D 2B 28 )?([\s()\-\./]+(
00000780: 5C 64 2B 29 29 3F 28 5B 5C 73 28 29 5C 2D 5C 2E \d+))?([\s()\-\.
00000790: 2F 5D 2B 28 5C 64 2B 29 29 3F 28 5B 5C 73 28 29 /]+(\d+))?([\s()
000007A0: 5C 2D 5C 2E 2F 5D 2B 28 5C 64 2B 29 29 3F 28 5B \-\./]+(\d+))?([
000007B0: 5C 73 28 29 5C 2D 5C 2E 2F 5D 2B 28 5C 64 2B 29 \s()\-\./]+(\d+)
000007C0: 29 3F 5C 73 2A 5B 58 78 5D 2B 28 5C 64 7B 31 2C )?\s*[Xx]+(\d{1,
000007D0: 31 35 7D 29 5B 5C 73 5D 2A 0D 0A 2B 24 31 24 33 15})[\s]*..+$1$3
000007E0: 24 35 24 37 24 39 24 31 31 24 31 33 24 31 35 24 $5$7$9$11$13$15$
000007F0: 31 37 24 31 39 24 32 31 3B 65 78 74 3D 24 32 32 17$19$21;ext=$22
00000800: 0D 0A 5C 2B 2B 28 5C 64 2B 29 28 5B 5C 73 28 29 ..\++(\d+)([\s()
00000810: 5C 2D 5C 2E 2F 5D 2B 28 5C 64 2B 29 29 3F 28 5B \-\./]+(\d+))?([
00000820: 5C 73 28 29 5C 2D 5C 2E 2F 5D 2B 28 5C 64 2B 29 \s()\-\./]+(\d+)
00000830: 29 3F 28 5B 5C 73 28 29 5C 2D 5C 2E 2F 5D 2B 28 )?([\s()\-\./]+(
00000840: 5C 64 2B 29 29 3F 28 5B 5C 73 28 29 5C 2D 5C 2E \d+))?([\s()\-\.
00000850: 2F 5D 2B 28 5C 64 2B 29 29 3F 28 5B 5C 73 28 29 /]+(\d+))?([\s()
00000860: 5C 2D 5C 2E 2F 5D 2B 28 5C 64 2B 29 29 3F 28 5B \-\./]+(\d+))?([
00000870: 5C 73 28 29 5C 2D 5C 2E 2F 5D 2B 28 5C 64 2B 29 \s()\-\./]+(\d+)
00000880: 29 3F 28 5B 5C 73 28 29 5C 2D 5C 2E 2F 5D 2B 28 )?([\s()\-\./]+(
00000890: 5C 64 2B 29 29 3F 28 5B 5C 73 28 29 5C 2D 5C 2E \d+))?([\s()\-\.
```

```
000008A0: 2F 5D 2B 28 5C 64 2B 29 29 3F 28 5B 5C 73 28 29 /]+(\d+))?([\s()
000008B0: 5C 2D 5C 2E 2F 5D 2B 28 5C 64 2B 29 29 3F 28 5B \-\./]+(\d+))?([
000008C0: 5C 73 28 29 5C 2D 5C 2E 2F 5D 2B 28 5C 64 2B 29 \s()\-\./]+(\d+)
000008D0: 29 3F 5B 5C 73 5D 2A 0D 0A 2B 24 31 24 33 24 35 )?[\s]*..+$1$3$5
000008E0: 24 37 24 39 24 31 31 24 31 33 24 31 35 24 31 37 $7$9$11$13$15$17
000008F0: 24 31 39 24 32 31 0D 0A 00                      $19$21...
```



Length (4 bytes): 0x0000085B specifies the length of the **Rules** field, in bytes.

Rules (variable): Normalization rules used by the Address Book Server to normalize the phone numbers in this file. This is a zero-terminated UTF-8 string, where the string contains one or more pairs of lines.

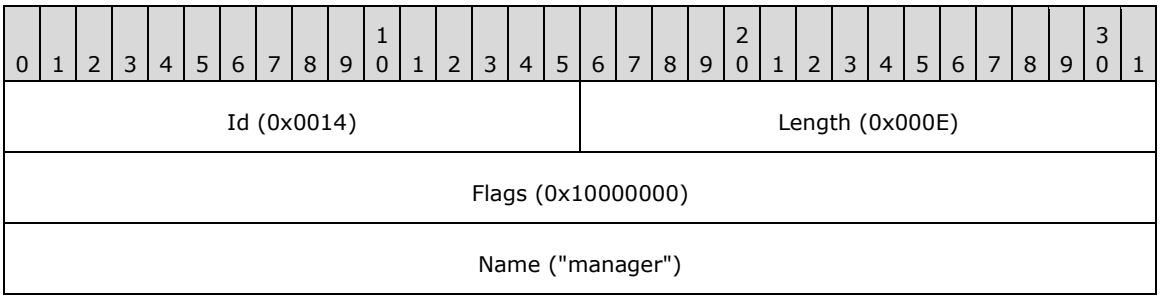
```
.*882[\s()\-\./]*8080.*[Xx]+[\s()\-\./]*(\d\d\d\d\d)$1;phone-
context=proseware.com.*882[\s()\-\./]*8080.*null\((\+\s*1)?\)[\s\-
\.]*\(?(\d\d\d)\s*\)?[\s()\-\./]*(\d\d\d)[\s()\-\./]*(\d\d\d\d)\s*[Xx]+[\s()\-
\./]*(\d\d\d\d\d)[\s()\-\./]*+1$2$3$4;ext=$5 (\+\s*1)?[\s\-
\.]*\(?(\d\d\d)\s*\)?[\s()\-\./]*(\d\d\d)[\s()\-
\./]*(\d\d\d\d)\s*EXT\s*(\d\d\d\d\d)+1$2$3$4;ext=$5 (\+\s*1)?[\s\-
\.]*\(?(\d\d\d)\s*\)?[\s()\-\./]*(\d\d\d)[\s()\-
\./]*(\d\d\d\d)\s*[xX]\s*(\d\d\d\d\d)+1$2$3$4;ext=$5 (\s*1)[\s\-
\.]*\(?\s*(\d\d\d)\s*\)?[\s()\-\./]*(\d\d\d)[\s()\-\./]*(\d\d\d\d)[\s]*+1$2$3$4
 (\+\s*1)?[\s\-\.]*\(?(\d\d\d)\s*\)?[\s()\-
\./]*70(\d\d\d\d\d)+1$270$3;ext=$370(\d)[\s()\-
\./]*(\d\d\d\d)\s*[Xx]+(\d\d\d\d\d)+142570$1$2;ext=$3 (\d\d\d)[\s()\-
\./]*(\d\d\d\d)\s*[Xx]+(\d\d\d\d\d)+1425$1$2;ext=$3 (\d\d\d)[\s()\-
\./]*(\d\d\d\d)+1425$1$2 [Xx]\s*(\d\d\d\d\d)$1;phone-
context=proseware.com\(?(\d\d)\s*\)?\s*[Xx]+\s*(\d\d\d\d\d)+1425$1$2;ext=$2011(\d+)([
\s()\-\./]+(\d+))?([\s()\-\./]+(\d+))?([\s()\-\./]+(\d+))?([\s()\-
\./]+(\d+))?([\s()\-\./]+(\d+))?([\s()\-\./]+(\d+))?([\s()\-
\./]+(\d+))?([\s()\-\./]+(\d+))?([\s()\-
\./]+(\d+))?\s*[Xx]+(\d{1,15})[\s]*+$1$3$5$7$9$11$13$15$17$19$21;ext=$22011(\d+)([\s(
)\-\./]+(\d+))?([\s()\-\./]+(\d+))?([\s()\-\./]+(\d+))?([\s()\-\./]+(\d+))?([\s()\-
\./]+(\d+))?([\s()\-\./]+(\d+))?([\s()\-\./]+(\d+))?([\s()\-
\./]+(\d+))?([\s()\-\./]+(\d+))?[\s]*+$1$3$5$7$9$11$13$15$17$19$21 (\d+)([\s()\-
\./]+(\d+))([\s()\-\./]+(\d+))([\s()\-\./]+(\d+))?([\s()\-\./]+(\d+))?([\s()\-
\./]+(\d+))?[\s]*+$1$3$5$7$9$11E164null\++(\d+)([\s()\-\./]+(\d+))?([\s()\-
\./]+(\d+))?([\s()\-\./]+(\d+))?([\s()\-\./]+(\d+))?([\s()\-\./]+(\d+))?([\s()\-
\./]+(\d+))?([\s()\-\./]+(\d+))?([\s()\-\./]+(\d+))?([\s()\-\./]+(\d+))?([\s()\-
\./]+(\d+))?\s*[Xx]+(\d{1,15})[\s]*+$1$3$5$7$9$11$13$15$17$19$21;ext=$22\++(\d+)([\s(
)\-\./]+(\d+))?([\s()\-\./]+(\d+))?([\s()\-\./]+(\d+))?([\s()\-\./]+(\d+))?([\s()\-
\./]+(\d+))?([\s()\-\./]+(\d+))?([\s()\-\./]+(\d+))?([\s()\-\./]+(\d+))?([\s()\-
\./]+(\d+))?([\s()\-\./]+(\d+))?[\s]*+$1$3$5$7$9$11$13$15$17$19$21
```

3.1.3 Attribute Structure

3.1.3.1 ABF_ATTRIBUTE manager

```
000008F0:                         0E 00 14 00 00 00 00          .......
```

```
00000900: 10 6D 61 6E 61 67 65 72 00                     .manager.
```



Length: 0x000E specifies the length of the remaining bytes in this structure.

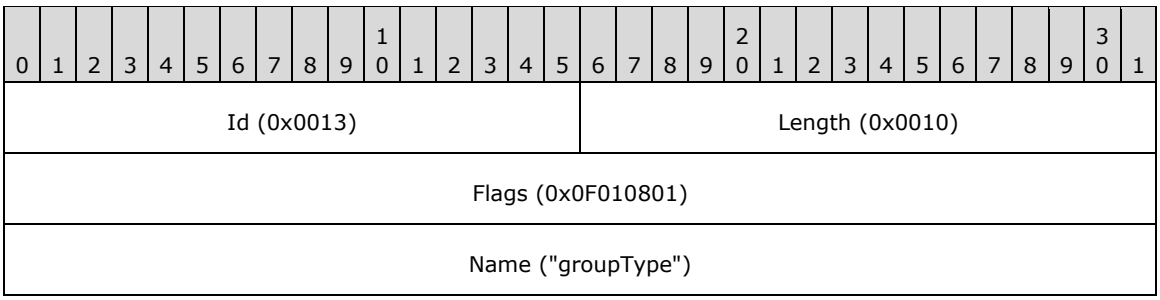
Id: 0x0014 specifies the identifier of this attribute as referenced by **ABF_CONTACT** structures and the **MaximumAttributeId** field of the **ABF_FULL_HEADER** and **ABF_DELTA_HEADER** structures.

Flags: 0x10000000 defines the type of attribute, how it is mapped by the client, and how it is processed by the Address Book Server. These flag settings correspond to the Boolean expression ABF_TYPE_STRING OR ABF_CLIENT_FIELD_MANAGER.

Name: Specifies that this attribute corresponds to the **manager** attribute associated with Active Directory user and contact objects.

3.1.3.2 ABF_ATTRIBUTE groupType

```
00000900:                         10 00 13 00 01 08 01          .......
00000910: 0F 67 72 6F 75 70 54 79 70 65 00               .groupType.
```



Length: 0x0010 specifies the length of the remaining bytes in this structure.

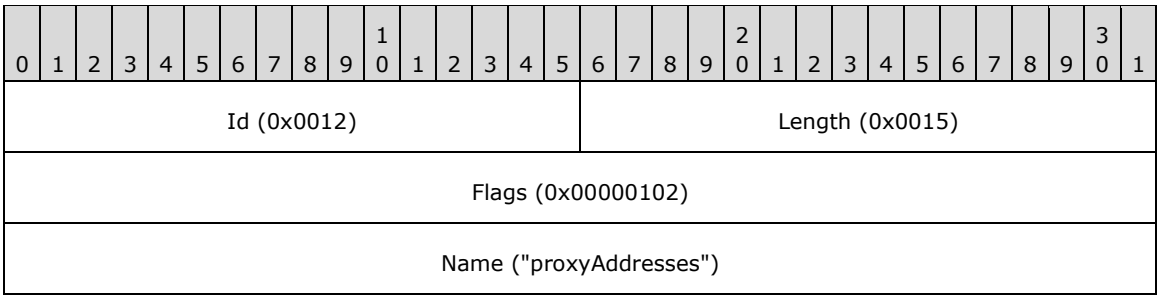
Id: 0x0013 specifies the identifier of this attribute as referenced by **ABF_CONTACT** structures and the **MaximumAttributeId** field of the **ABF_FULL_HEADER** and **ABF_DELTA_HEADER** structures.

Flags: 0x0F010801 defines the type of attribute, how it is mapped by the client, and how it is processed by the Address Book Server. These flag settings correspond to the Boolean expression ABF_TYPE_BINARY OR ABF_ATTRIBUTE_FLAGS_REQUIRED OR ABF_ATTRIBUTE_FLAGS_GROUPTYPE OR ABF_CLIENT_FIELD_GROUPTYPE.

Name: Specifies that this attribute corresponds to the **groupType** attribute associated with Active Directory group objects.

3.1.3.3 ABF_ATTRIBUTE proxyAddresses

```
00000910:                                 15 00 12 00 02           .....
00000920: 01 00 00 70 72 6F 78 79 41 64 64 72 65 73 73 65 ...proxyAddresse
00000930: 73 00                                          s.
```



Length: 0x0015 specifies the length of the remaining bytes in this structure.

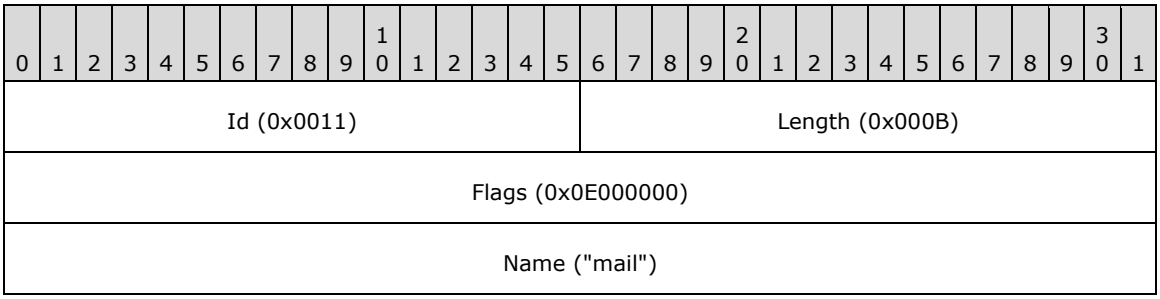
Id: 0x0012 specifies the identifier of this attribute as referenced by **ABF_CONTACT** structures and the **MaximumAttributeId** field of the **ABF_FULL_HEADER** and **ABF_DELTA_HEADER** structures.

Flags: 0x00000102 defines the type of attribute, how it is mapped by the client, and how it is processed by the Address Book Server. These flag settings correspond to the Boolean expression ABF_TYPE_STRING_PROXYADDRESS OR ABF_ATTRIBUTE_FLAGS_RESERVED OR ABF_CLIENT_FIELD_PROXYADDRESSES.

Name: Specifies that this attribute corresponds to the **proxyAddress**es attribute associated with Active Directory user, contact and group objects.

3.1.3.4 ABF_ATTRIBUTE mail

```
00000930:      0B 00 11 00 00 00 00 0E 6D 61 69 6C 00      ........mail.
```



Length: 0x000B specifies the length of the remaining bytes in this structure.

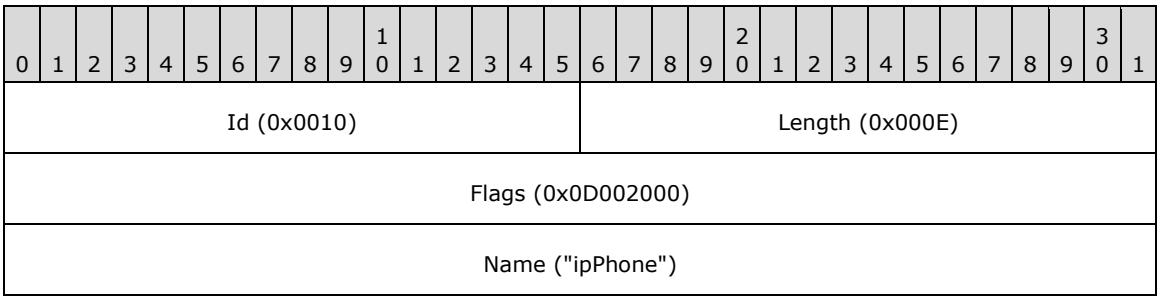
Id: 0x0011 specifies the identifier of this attribute as referenced by **ABF_CONTACT** structures and the **MaximumAttributeId** field of the **ABF_FULL_HEADER** and **ABF_DELTA_HEADER** structures.

Flags: 0x0E000000 defines the type of attribute, how it is mapped by the client, and how it is processed by the Address Book Server. These flag settings correspond to the Boolean expression ABF_TYPE_STRING OR ABF_CLIENT_FIELD_MAIL.

Name: Specifies that this attribute corresponds to the **mail** attribute associated with Active Directory user and contact objects.

3.1.3.5 ABF_ATTRIBUTE ipPhone

```
00000930:                                                 0E                .
00000940: 00 10 00 00 20 00 0D 69 70 50 68 6F 6E 65 00     .......ipPhone.
```



Length: 0x000E specifies the length of the remaining bytes in this structure.

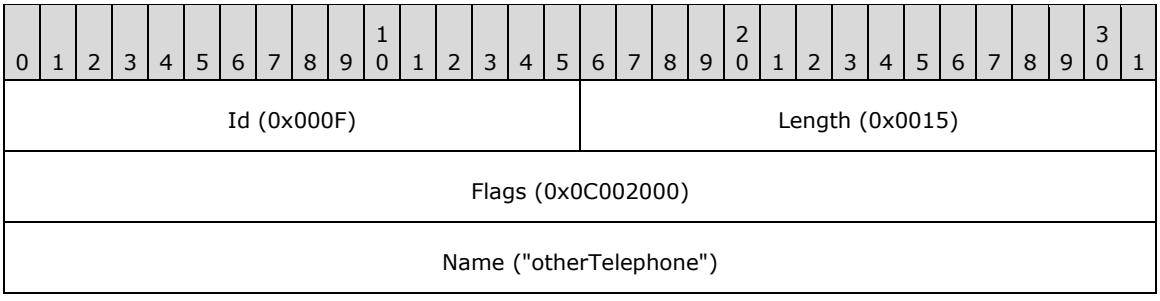
Id: 0x0010 specifies the identifier of this attribute as referenced by **ABF_CONTACT** structures and the **MaximumAttributeId** field of the **ABF_FULL_HEADER** and **ABF_DELTA_HEADER** structures.

Flags: 0x0D002000 defines the type of attribute, how it is mapped by the client, and how it is processed by the Address Book Server. These flag settings correspond to the Boolean expression ABF_TYPE_STRING OR ABF_NORMALIZATION_CONTROL_PROXYADDRESS OR ABF_CLIENT_FIELD_IPPHONE.

Name: Specifies that this attribute corresponds to the **ipPhone** attribute associated with Active Directory user and contact objects.

3.1.3.6 ABF_ATTRIBUTE otherTelephone

```
00000940:                                                 15                .
00000950: 00 0F 00 00 20 00 0C 6F 74 68 65 72 54 65 6C 65 .......otherTele
00000960: 70 68 6F 6E 65 00                                  phone.
```



Length: 0x0015 specifies the length of the remaining bytes in this structure.

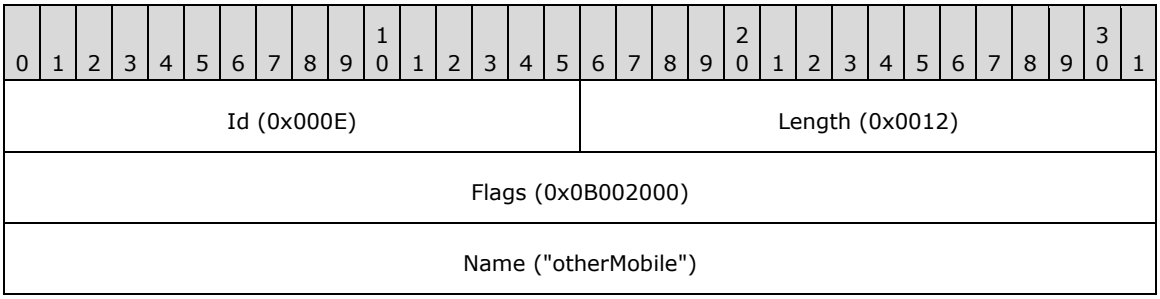
Id: 0x000F specifies the identifier of this attribute as referenced by **ABF_CONTACT** structures and the **MaximumAttributeId** field of the **ABF_FULL_HEADER** and **ABF_DELTA_HEADER** structures.

Flags: 0x0C002000 defines the type of attribute, how it is mapped by the client, and how it is processed by the Address Book Server. These flag settings correspond to the Boolean expression ABF_TYPE_STRING OR ABF_NORMALIZATION_CONTROL_PROXYADDRESS OR ABF_CLIENT_FIELD_OTHERTELEPHONE

Name: Specifies that this attribute corresponds to the **otherTelephone** attribute associated with Active Directory user and contact objects.

3.1.3.7 ABF_ATTRIBUTE otherMobile

```
00000960:                   12 00 0E 00 00 20 00 0B 6F 74       ........ot
00000970: 68 65 72 4D 6F 62 69 6C 65 00                   herMobile.
```



Length: 0x0012 specifies the length of the remaining bytes in this structure.

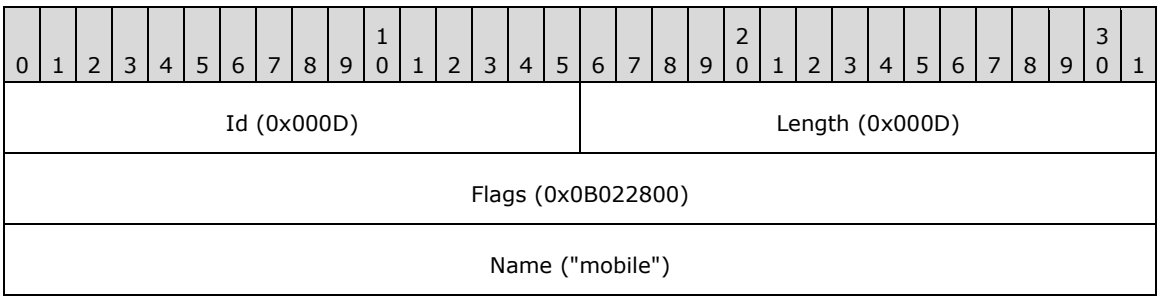
Id: 0x000E specifies the identifier of this attribute as referenced by **ABF_CONTACT** structures and the **MaximumAttributeId** field of the **ABF_FULL_HEADER** and **ABF_DELTA_HEADER** structures.

Flags: 0x0B002000 defines the type of attribute, how it is mapped by the client, and how it is processed by the Address Book Server. These flag settings correspond to the Boolean expression ABF_TYPE_STRING OR ABF_NORMALIZATION_CONTROL_PROXYADDRESS OR ABF_CLIENT_FIELD_MOBILE.

Name: Specifies that this attribute corresponds to the **otherMobile** attribute associated with Active Directory user and contact objects.

3.1.3.8 ABF_ATTRIBUTE mobile

```
00000970:                               0D 00 0D 00 00 28           .....(
00000980: 02 0B 6D 6F 62 69 6C 65 00                      ..mobile.
```



Length: 0x000D specifies the length of the remaining bytes in this structure.

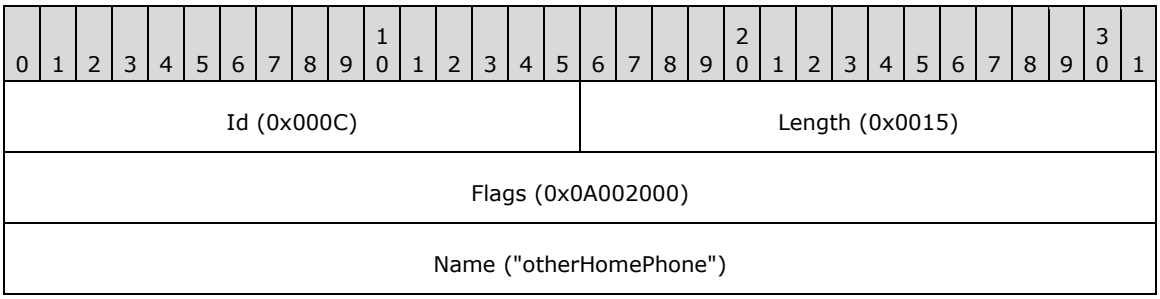
Id: 0x000D specifies the identifier of this attribute as referenced by **ABF_CONTACT** structures and the **MaximumAttributeId** field of the **ABF_FULL_HEADER** and **ABF_DELTA_HEADER** structures.

Flags: 0x0B022800 defines the type of attribute, how it is mapped by the client, and how it is processed by the Address Book Server. These flag settings correspond to the Boolean expression ABF_TYPE_STRING OR ABF_ATTRIBUTE_FLAGS_REQUIRED OR ABF_NORMALIZATION_CONTROL_PROXYADDRESS OR ABF_ATTRIBUTE_FLAGS_DEVICE OR ABF_CLIENT_FIELD_MOBILE.

Name: Specifies that this attribute corresponds to the **mobile** attribute associated with Active Directory user and contact objects.

3.1.3.9 ABF_ATTRIBUTE otherHomePhone

```
00000980:                           15 00 0C 00 00 20 00          .......
00000990: 0A 6F 74 68 65 72 48 6F 6D 65 50 68 6F 6E 65 00 .otherHomePhone.
000009A0:
```



Length: 0x0015 specifies the length of the remaining bytes in this structure.

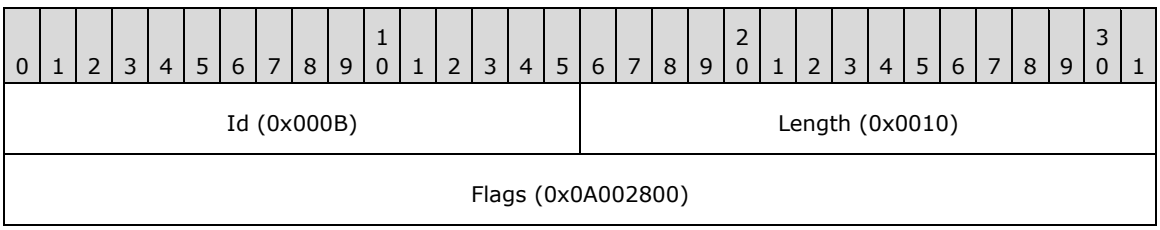
Id: 0x000C specifies the identifier of this attribute as referenced by **ABF_CONTACT** structures and the **MaximumAttributeId** field of the **ABF_FULL_HEADER** and **ABF_DELTA_HEADER** structures.

Flags: 0x0A002000 defines the type of attribute, how it is mapped by the client, and how it is processed by the Address Book Server. These flag settings correspond to the Boolean expression ABF_TYPE_STRING OR ABF_NORMALIZATION_CONTROL_PROXYADDRESS OR ABF_CLIENT_FIELD_HOMENUMBER.

Name: Specifies that this attribute corresponds to the **otherHomePhone** attribute associated with Active Directory user and contact objects.

3.1.3.10 ABF_ATTRIBUTE homePhone

```
000009a0: 10 00 0b 00 00 28 00 0a 68 6f 6d 65 50 68 6f 6e .....(..homePhon
000009b0: 65 00                                           e.
```



Name ("homePhone")

Length: 0x0010 specifies the length of the remaining bytes in this structure.

Id: 0x000B specifies the identifier of this attribute as referenced by **ABF_CONTACT** structures and the **MaximumAttributeId** field of the **ABF_FULL_HEADER** and **ABF_DELTA_HEADER** structures.

Flags: 0x0A002800 defines the type of attribute, how it is mapped by the client, and how it is processed by the Address Book Server. These flag settings correspond to the Boolean expression ABF_TYPE_STRING OR ABF_ATTRIBUTE_FLAGS_REQUIRED OR ABF_NORMALIZATION_CONTROL_PROXYADDRESS OR ABF_CLIENT_FIELD_HOMENUMBER.

Name: Specifies that this attribute corresponds to the **homePhone** attribute associated with Active Directory user and contact objects.

3.1.3.11 ABF_ATTRIBUTE telephoneNumber

```
000009B0:       16 00 0A 00 00 28 02 09 74 65 6C 65 70 68   .....(..teleph
000009C0: 6F 6E 65 4E 75 6D 62 65 72 00                    oneNumber.
```

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
Id (0x000A)																Length (0x0016)															
Flags (0x09022800)																															
Name ("telephoneNumber")																															

Length: 0x0016 specifies the length of the remaining bytes in this structure.

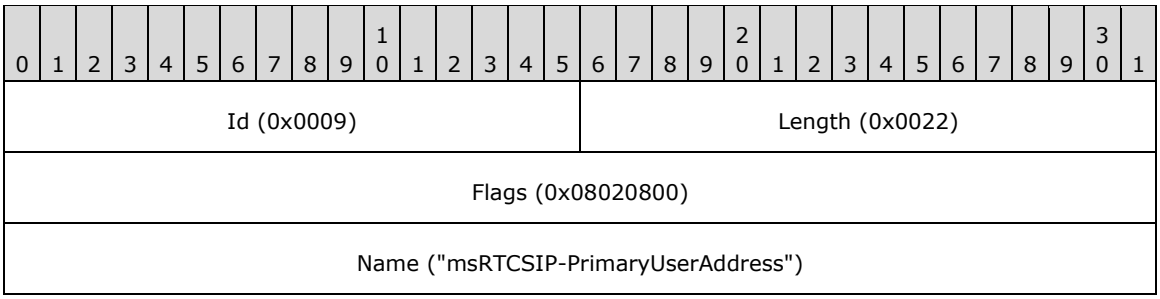
Id: 0x000A specifies the identifier of this attribute as referenced by **ABF_CONTACT** structures and the **MaximumAttributeId** field of the **ABF_FULL_HEADER** and **ABF_DELTA_HEADER** structures.

Flags: 0x09022800 defines the type of attribute, how it is mapped by the client, and how it is processed by the Address Book Server. These flag settings correspond to the Boolean expression ABF_TYPE_STRING OR ABF_ATTRIBUTE_FLAGS_REQUIRED OR ABF_NORMALIZATION_CONTROL_PROXYADDRESS OR ABF_ATTRIBUTE_FLAGS_DEVICE OR ABF_CLIENT_FIELD_TELEPHONENUMBER.

Name: Specifies that this attribute corresponds to the **telephoneNumber** attribute associated with Active Directory user and contact objects.

3.1.3.12 ABF_ATTRIBUTE msRTCSIP-PrimaryUserAddress

```
000009C0:                               22 00 09 00 00 08          ".....
000009D0: 02 08 6D 73 52 54 43 53 49 50 2D 50 72 69 6D 61 ..msRTCSIP-Prima
000009E0: 72 79 55 73 65 72 41 64 64 72 65 73 73 00       ryUserAddress.
```



Length: 0x0022 specifies the length of the remaining bytes in this structure.

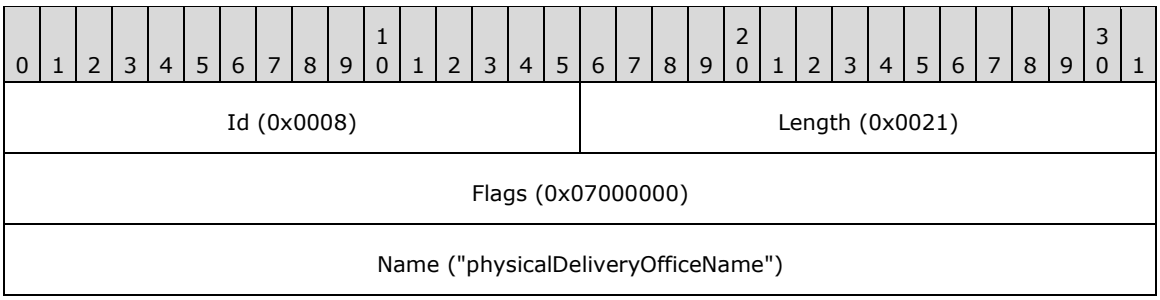
Id: 0x0009 specifies the identifier of this attribute as referenced by **ABF_CONTACT** structures and the **MaximumAttributeId** field of the **ABF_FULL_HEADER** and **ABF_DELTA_HEADER** structures.

Flags: 0x08020800 defines the type of attribute, how it is mapped by the client, and how it is processed by the Address Book Server. These flag settings correspond to the Boolean expression ABF_TYPE_STRING OR ABF_ATTRIBUTE_FLAGS_REQUIRED OR ABF_ATTRIBUTE_FLAGS_DEVICE OR ABF_CLIENT_FIELD_MSRTCSIP_PRIMARYUSERADDRESS.

Name: Specifies that this attribute corresponds to the **msRTCSIP-PrimaryUserAddress** attribute associated with Active Directory user and contact objects.

3.1.3.13 ABF_ATTRIBUTE physicalDeliveryOfficeName

```
000009E0:                                           21 00              !.
000009F0: 08 00 00 00 00 07 70 68 79 73 69 63 61 6C 44 65 ......physicalDe
00000A00: 6C 69 76 65 72 79 4F 66 66 69 63 65 4E 61 6D 65 liveryOfficeName
00000A10: 00                                              .
```



Length: 0x0021 specifies the length of the remaining bytes in this structure.

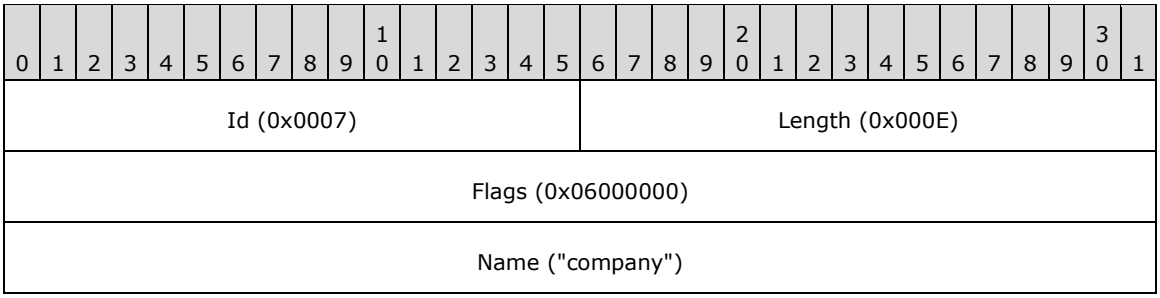
Id: 0x0008 specifies the identifier of this attribute as referenced by **ABF_CONTACT** structures and the **MaximumAttributeId** field of the **ABF_FULL_HEADER** and **ABF_DELTA_HEADER** structures.

Flags: 0x07000000 defines the type of attribute, how it is mapped by the client, and how it is processed by the Address Book Server. These flag settings correspond to the Boolean expression ABF_TYPE_STRING OR ABF_CLIENT_FIELD_PHYSICALDELIVERYOFFICENAME.

Name: Specifies that this attribute corresponds to the **physicalDeliveryOfficeName** attribute associated with Active Directory user and contact objects.

3.1.3.14 ABF_ATTRIBUTE company

```
00000A10:    0E 00 07 00 00 00 00 06 63 6F 6D 70 61 6E 79  ........company
00000A20: 00                                               .
```



Length: 0x000E specifies the length of the remaining bytes in this structure.

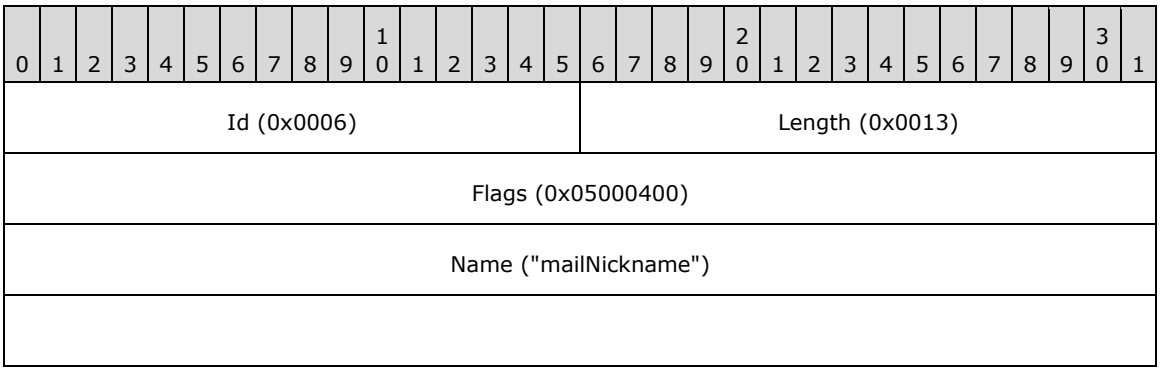
Id: 0x0007 specifies the identifier of this attribute as referenced by **ABF_CONTACT** structures and the **MaximumAttributeId** field of the **ABF_FULL_HEADER** and **ABF_DELTA_HEADER** structures.

Flags: 0x06000000 defines the type of attribute, how it is mapped by the client, and how it is processed by the Address Book Server. These flag settings correspond to the Boolean expression ABF_TYPE_STRING OR ABF_CLIENT_FIELD_COMPANY.

Name: Specifies that this attribute corresponds to the **company** attribute associated with Active Directory user and contact objects.

3.1.3.15 ABF_ATTRIBUTE mailNickname

```
00000A20:    13 00 06 00 00 04 00 05 6D 61 69 6C 4E 69 63  ........mailNic
00000A30: 6B 6E 61 6D 65 00                                kname.
```



Length: 0x0013 specifies the length of the remaining bytes in this structure.

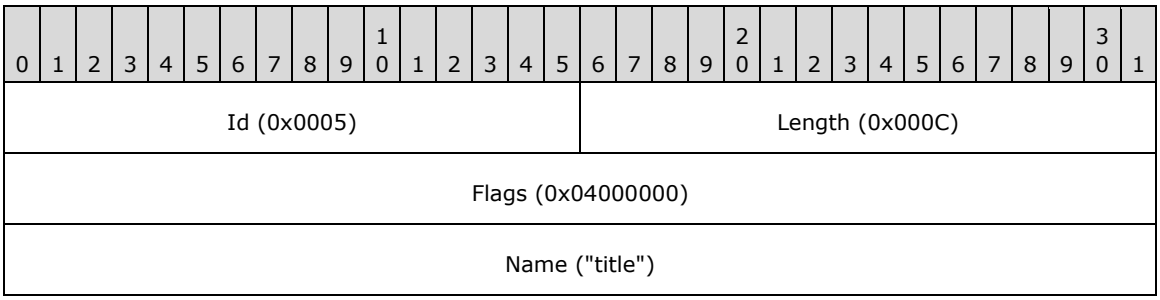
Id: 0x0006 specifies the identifier of this attribute as referenced by **ABF_CONTACT** structures and the **MaximumAttributeId** field of the **ABF_FULL_HEADER** and **ABF_DELTA_HEADER** structures.

Flags: 0x05000400 defines the type of attribute, how it is mapped by the client, and how it is processed by the Address Book Server. These flag settings correspond to the Boolean expression ABF_TYPE_STRING OR ABF_ATTRIBUTE_FLAGS_EMAIL OR ABF_CLIENT_FIELD_MAILNICKNAME.

Name: Specifies that this attribute corresponds to the **mailNickname** attribute associated with Active Directory user, contact and group objects.

3.1.3.16 ABF_ATTRIBUTE title

```
00000A30:                   0C 00 05 00 00 00 00 04 74 69       ........ti
00000A40: 74 6C 65 00                                       tle.
```



Length: 0x000C specifies the length of the remaining bytes in this structure.

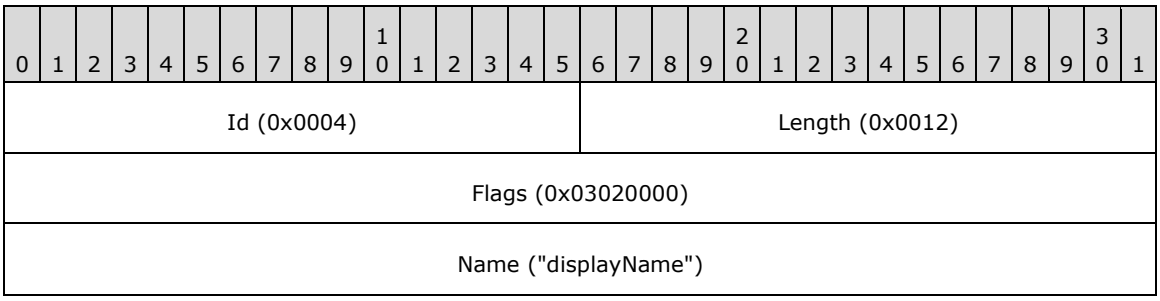
Id: 0x0005 specifies the identifier of this attribute as referenced by **ABF_CONTACT** structures and the **MaximumAttributeId** field of the **ABF_FULL_HEADER** and **ABF_DELTA_HEADER** structures.

Flags: 0x04000000 defines the type of attribute, how it is mapped by the client, and how it is processed by the Address Book Server. These flag settings correspond to the Boolean expression ABF_TYPE_STRING OR ABF_CLIENT_FIELD_TITLE.

Name: Specifies that this attribute corresponds to the **title** attribute associated with Active Directory user and contact objects.

3.1.3.17 ABF_ATTRIBUTE displayName

```
00000A40:            12 00 04 00 00 00 02 03 64 69 73 70     ........disp
00000A50: 6C 61 79 4E 61 6D 65 00                           layName.
```



Length: 0x0012 specifies the length of the remaining bytes in this structure.

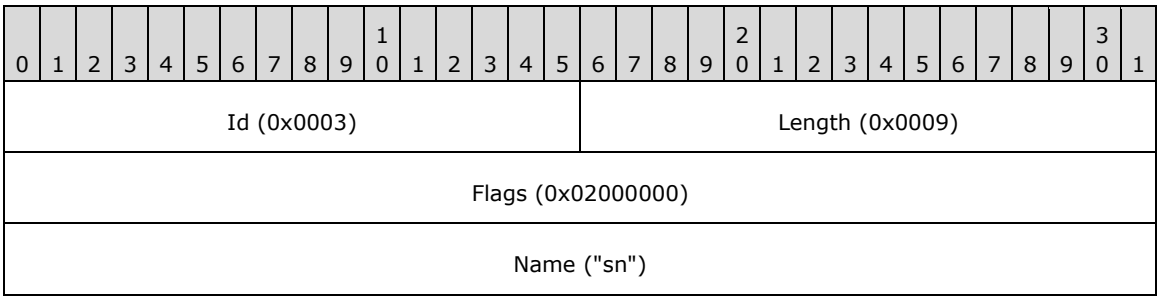
Id: 0x0004 specifies the identifier of this attribute as referenced by **ABF_CONTACT** structures and the **MaximumAttributeId** field of the **ABF_FULL_HEADER** and **ABF_DELTA_HEADER** structures.

Flags: 0x03020000 defines the type of attribute, how it is mapped by the client, and how it is processed by the Address Book Server. These flag settings correspond to the Boolean expression ABF_TYPE_STRING OR ABF_ATTRIBUTE_FLAGS_DEVICE OR ABF_CLIENT_FIELD_DISPLAYNAME

Name: Specifies that this attribute corresponds to the **displayName** attribute associated with Active Directory user, contact and group objects.

3.1.3.18 ABF_ATTRIBUTE sn

```
00000A50:                         09 00 03 00 00 00 00 02         ........
00000A60: 73 6E 00                                        sn.
```



Length: 0x0009 specifies the length of the remaining bytes in this structure.

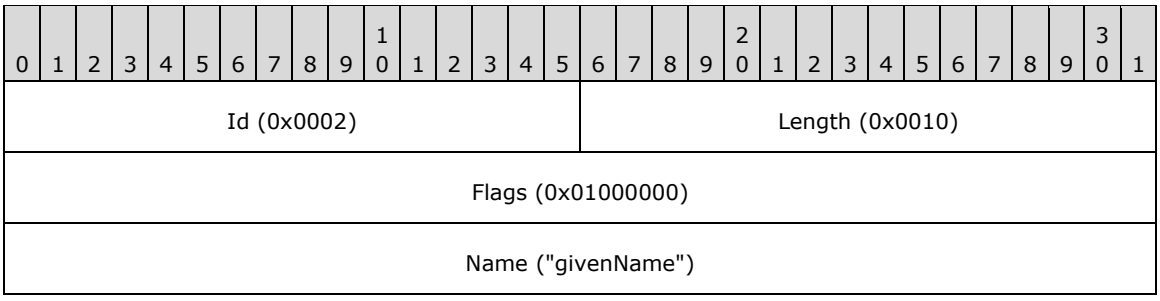
Id: 0x0003 specifies the identifier of this attribute as referenced by **ABF_CONTACT** structures and the **MaximumAttributeId** field of the **ABF_FULL_HEADER** and **ABF_DELTA_HEADER** structures.

Flags: 0x02000000 defines the type of attribute, how it is mapped by the client, and how it is processed by the Address Book Server. These flag settings correspond to the Boolean expression ABF_TYPE_STRING OR ABF_CLIENT_FIELD_SN.

Name: Specifies that this attribute corresponds to the **sn** attribute associated with Active Directory user and contact objects.

3.1.3.19 ABF_ATTRIBUTE givenName

```
00000A60:          10 00 02 00 00 00 00 01 67 69 76 65 6E    ........given
00000A70: 4E 61 6D 65 00                                  Name.
```



Length: 0x0010 specifies the length of the remaining bytes in this structure.

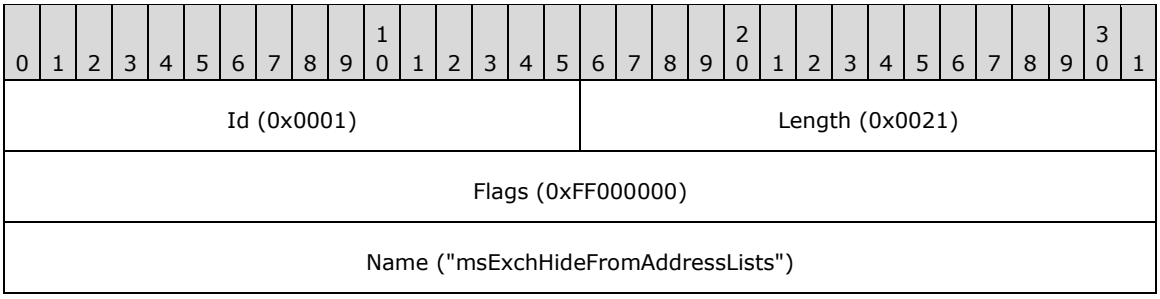
Id: 0x0002 specifies the identifier of this attribute as referenced by **ABF_CONTACT** structures and the **MaximumAttributeId** field of the **ABF_FULL_HEADER** and **ABF_DELTA_HEADER** structures.

Flags: 0x01000000 defines the type of attribute, how it is mapped by the client, and how it is processed by the Address Book Server. These flag settings correspond to the Boolean expression ABF_TYPE_STRING OR ABF_CLIENT_FIELD_GIVENNAME.

Name: Specifies that this attribute corresponds to the **givenName** attribute associated with Active Directory user and contact objects.

3.1.3.20 ABF_ATTRIBUTE msExchHideFromAddressLists

```
00000A70:                21 00 01 00 00 00 FF 6D 73 45      !.......msE
00000A80: 78 63 68 48 69 64 65 46 72 6F 6D 41 64 64 72 65 xchHideFromAddre
00000A90: 73 73 4C 69 73 74 73 00                         ssLists.
```



Length: 0x0021 specifies the length of the remaining bytes in this structure.

Id: 0x0001 specifies the identifier of this attribute as referenced by **ABF_CONTACT** structures and the **MaximumAttributeId** field of the **ABF_FULL_HEADER** and **ABF_DELTA_HEADER** structures.

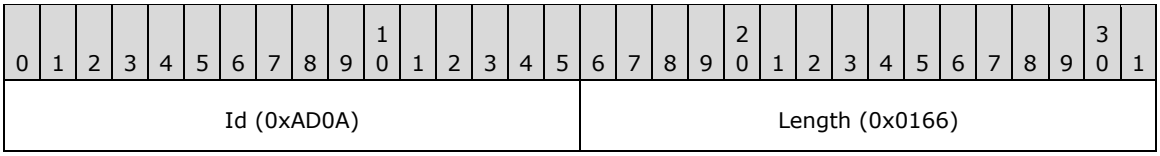
Flags: 0xFF000000 defines the type of attribute, how it is mapped by the client, and how it is processed by the Address Book Server. These flag settings correspond to the Boolean expression ABF_TYPE_STRING OR ABF_CLIENT_FIELD_IGNORE.

Name: Specifies that this attribute corresponds to the **msExchHideFromAddressLists** attribute associated with Active Directory user, contact and group objects.

3.1.4 ABSUser1 Contact

3.1.4.1 ABF_CONTACT

```
00000A90:                         66 01 0A AD 36 8C 97 5E         f...6..^
00000AA0: DD 46 8D 5B 25 51 40 C5 2B 00 12 00             .F.[%Q@.+...
```



0x5E978C36	
0x5B8D46DD	
0xC5405125	
NumberOfAttributes (0x0012)	0x002B

Length: 0x0166 specifies the length of the remaining bytes in the structure.

Id: {8c36ad0a-5e97-46dd-8d5b-255140c52b00} is the **objectGUID** value associated with ABSUser1 in the sample Active Directory data.

NumberOfAttributes: 0x0012 specifies the number of **ABF_CONTACT_ATTRIBUTE** structures that follow this structure.

3.1.4.2 ABF_CONTACT_ATTRIBUTE mail

```
00000AA0:                                     11 41 42 53            .ABS
00000AB0: 55 73 65 72 31 40 75 72 74 65 73 74 2E 63 6F 6D User1@urtest.com
00000AC0: 00                                                 .
```

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
"ABSUser1@urtest.com"																Id (0x11)															

Id: 0x11 corresponds to the **mail ABF_ATTRIBUTE** structure.

Data: "ABSUser1@urtest.com" is a null-terminated string that is the value of the **mail** attribute for this contact.

3.1.4.3 ABF_CONTACT_ATTRIBUTE otherTelephone

```
00000AC0:    0F 35 35 35 2D 35 33 33 2D 34 33 31 32 00     .555-533-4312.
```

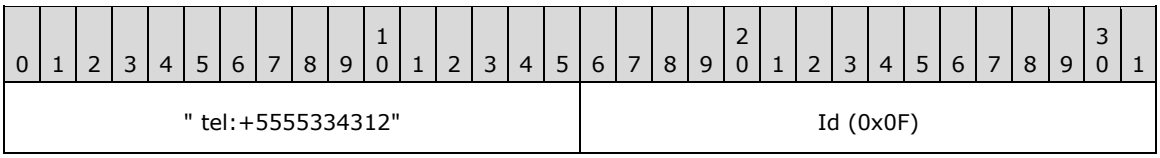
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
"555-533-4312"																Id (0x0F)															

Id: 0x0F corresponds to the **otherTelephone ABF_ATTRIBUTE** structure.

Data: "555-533-4312" is a null-terminated string that is the value of the **otherTelephone** attribute for this contact.

3.1.4.4 ABF_CONTACT_ATTRIBUTE otherTelephone, normalized

```
00000AC0:                                                 0F                .
00000AD0: 74 65 6C 3A 2B 35 35 35 35 33 33 34 33 31 32 00 tel:+5555334312.
00000AE0:
```

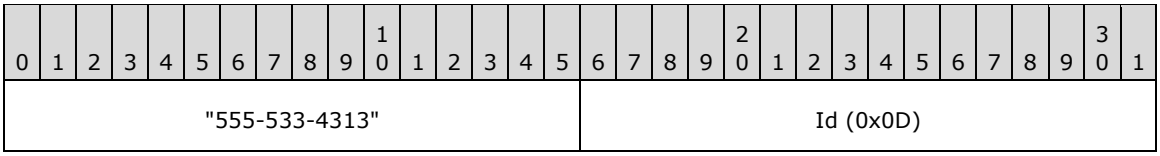


Id: 0x0F corresponds to the **otherTelephone ABF_ATTRIBUTE** structure.

Data: "tel:+5555334312" is a null-terminated string that is the value of the **otherTelephone** attribute for this contact. This attribute value is not present in Active Directory, because this is the normalized version of the actual value of the **otherTelephone** attribute stored in Active Directory that was reported in the previous **ABF_CONTACT_ATTRIBUTE** structure.

3.1.4.5 ABF_CONTACT_ATTRIBUTE mobile

```
00000AE0: 0D 35 35 35 2D 35 33 33 2D 34 33 31 33 00       .555-533-4313.
```

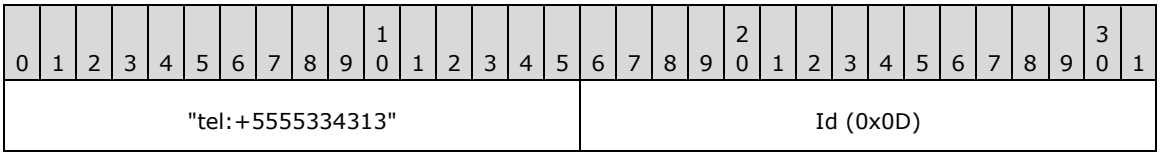


Id: 0x0D corresponds to the **mobile ABF_ATTRIBUTE** structure.

Data: "555-533-4313" is a null-terminated string that is the value of the **mobile** attribute for this contact.

3.1.4.6 ABF_CONTACT_ATTRIBUTE mobile, normalized

```
00000AE0:                                           0D 74             .t
00000AF0: 65 6C 3A 2B 35 35 35 35 33 33 34 33 31 33 00    el:+5555334313.
```

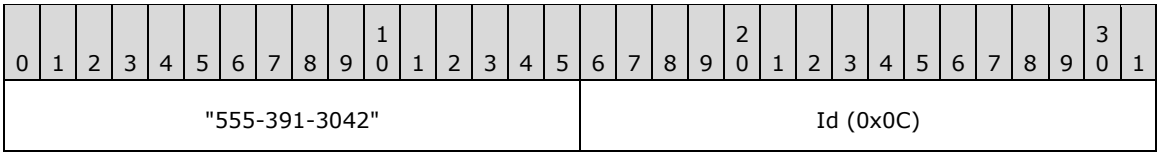


Id: 0x0D corresponds to the **mobile ABF_ATTRIBUTE** structure.

Data: "tel:+5555334313" is a null-terminated string that is the normalized version of the **mobile** attribute for this contact.

3.1.4.7 ABF_CONTACT_ATTRIBUTE otherHomePhone

```
00000AF0:                                                 0C                .
00000B00: 35 35 35 2D 33 39 31 2D 33 30 34 32 00             555-391-3042.
```

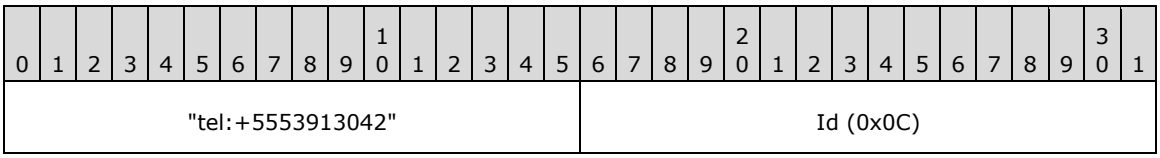


Id: 0x0C corresponds to the **otherHomePhone ABF_ATTRIBUTE** structure.

Data: "555-391-3042" is a null-terminated string that is the value of the "otherHomePhone" attribute for this contact.

3.1.4.8 ABF_CONTACT_ATTRIBUTE otherHomePhone, normalized

```
00000B00:                                       0C 74 65             .te
00000B10: 6C 3A 2B 35 35 35 33 39 31 33 30 34 32 00       l:+5553913042.
```

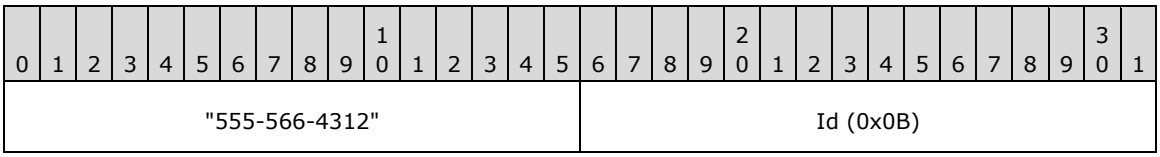


Id: 0x0C corresponds to the **otherHomePhone ABF_ATTRIBUTE** structure.

Data: "tel:+5553913042" is a null-terminated string that is the normalized version of the "otherHomePhone" attribute for this contact.

3.1.4.9 ABF_CONTACT_ATTRIBUTE homePhone

```
00000B10:                                          0B 35              .5
00000B20: 35 35 2D 35 36 36 2D 34 33 31 32 00             55-566-4312.
```



Id: 0x0B corresponds to the **homePhone ABF_ATTRIBUTE** structure.

Data: "555-566-4312" is a null-terminated string that is the value of the **homePhone** attribute for this contact.

3.1.4.10 ABF_CONTACT_ATTRIBUTE homePhone, normalized

```
00000B20:                                    0B 74 65 6C             .tel
00000B30: 3A 2B 35 35 35 35 36 36 34 33 31 32 00          :+5555664312.
```

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
"tel:+5555664312"																Id (0x0B)															

Id: 0x0B corresponds to the **homePhone ABF_ATTRIBUTE** structure.

Data: "tel:+5555664312" is a null-terminated string that is the normalized value of the **homePhone** attribute for this contact.

3.1.4.11 ABF_CONTACT_ATTRIBUTE telephoneNumber

```
00000B30:                                       0A 35 35              .55
00000B40: 35 20 33 39 31 20 33 32 32 34 00               5 391 3224.
```

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
"555 391 3224"																Id (0x0A)															

Id: 0x0A corresponds to the **telephoneNumber ABF_ATTRIBUTE** structure.

Data: "555 391 3224" is a null-terminated string that is the value of the "telephoneNumber" attribute for this contact.

3.1.4.12 ABF_CONTACT_ATTRIBUTE telephoneNumber, normalized

```
00000B40:                                 0A 74 65 6C 3A           .tel:
00000B50: 2B 35 35 35 33 39 31 33 32 32 34 00            +5553913224.
```

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
"tel:+5553913224"																Id (0x0A)															

Id: 0x0A corresponds to the **telephoneNumber ABF_ATTRIBUTE** structure.

Data: "tel:+5553913224" is a null-terminated string that is the normalized value of the **telephoneNumber** attribute for this contact.

3.1.4.13 ABF_CONTACT_ATTRIBUTE msRTCSIP-PrimaryUserAddress

```
00000B50:                                    09 73 69 70              .sip
00000B60: 3A 41 42 53 55 73 65 72 31 40 75 72 74 65 73 74 :ABSUser1@urtest
00000B70: 2E 72 74 6D 70 2E 73 65 6C 66 68 6F 73 74 2E 63 .rtmp.selfhost.c
00000B80: 6F 72 70 2E 70 72 6F 73 65 77 61 72 65 2E 63 6F orp.proseware.co
00000B90: 6D 00                                          m.
```

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
"sip:ABSUser1@urtest.rtmp.selfhost.corp.proseware.com"																Id (0x09)															

Id: 0x09 corresponds to the **msRTCSIP-PrimaryUserAddress ABF_ATTRIBUTE** structure.

Data: "sip:ABSUser1@urtest.rtmp.selfhost.corp.proseware.com" is a null-terminated string that is the value of the **msRTCSIP-PrimaryUserAddress** attribute for this contact.

3.1.4.14 ABF_CONTACT_ATTRIBUTE physicalDeliveryOfficeName

```
00000B90:       08 31 32 33 34 35 00                        .12345.
```

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
"12345"																Id (0x08)															

Id: 0x08 corresponds to the **physicalDeliveryOfficeName ABF_ATTRIBUTE** structure.

Data: "12345" is a null-terminated string that is the value of the **physicalDeliveryOfficeName** attribute for this contact.

3.1.4.15 ABF_CONTACT_ATTRIBUTE company

```
00000B90:                           07 54 65 73 74 43 6F          .TestCo
00000BA0: 6D 70 61 6E 79 00                                 mpany.
```

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
"TestCompany"																Id (0x07)															

Id: 0x07 corresponds to the **company ABF_ATTRIBUTE** structure.

Data: "TestCompany" is a null-terminated string that is the value of the **company** attribute for this contact.

3.1.4.16 ABF_CONTACT_ATTRIBUTE title

```
00000BA0:                   05 44 65 76 65 6C 6F 70 6D 65       .Developme
00000BB0: 6E 74 20 4D 61 6E 61 67 65 72 00                  nt.Manager.
```

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
"Development Manager"																Id (0x05)															

Id: 0x05 corresponds to the **title ABF_ATTRIBUTE** structure.

Data: "Development Manager" is a null-terminated string that is the value of the **title** attribute for this contact.

3.1.4.17 ABF_CONTACT_ATTRIBUTE displayName

```
00000BB0:                                  04 41 42 53 55            .ABSU
00000BC0: 73 65 72 31 5F 64 69 73 70 6C 61 79 6E 61 6D 65 ser1 displayname
00000BD0: 5F 63 68 61 6E 67 65 64 00                      _changed.
```

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
"ABSUser1_displayname_changed"																Id (0x04)															

Id: 0x04 corresponds to the **displayName ABF_ATTRIBUTE** structure.

Data: "ABSUser1_displayname_changed" is a null-terminated string that is the value of the **displayName** attribute for this contact.

3.1.4.18 ABF_CONTACT_ATTRIBUTE sn

```
00000BD0: 03 41 42 53 55 73 65 .ABSUse
00000BE0: 72 31 5F 6C 61 73 74 6E 61 6D 65 00 r1_lastname.
```

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
"ABSUser1_lastname"																Id (0x03)															

Id: 0x03 corresponds to the **sn ABF_ATTRIBUTE** structure.

Data: "ABSUser1_lastname" is a null-terminated string that is the value of the **sn** attribute for this contact.

3.1.4.19 ABF_CONTACT_ATTRIBUTE givenName

```
00000BE0: 02 41 42 53 .ABS
00000BF0: 55 73 65 72 31 5F 66 69 72 73 74 6E 61 6D 65 00 User1_firstname.
00000C00:
```

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
"ABSUser1_firstname"																Id (0x02)															

Id: 0x02 corresponds to the **givenName ABF_ATTRIBUTE** structure.

Data: "ABSUser1_firstname" is a null-terminated string that is the value of the **givenName** attribute for this contact.

3.1.5 ABSUser5 Contact

3.1.5.1 ABF_CONTACT

```
00000C00: A4 00 0B 41 E8 E1 C4 C8 22 40 93 BA B2 14 5E D6 ...A....."@....^.
00000C10: D1 34 06 00                                     .4..
```

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
Id (0x410B)																Length (0x00A4)															
0xC8C4E1E8																															
0xBA934022																															
0xD65E14B2																															
NumberOfAttributes (0x0006)																0x34D1															

Length: 0x00A4 specifies the length of the remaining bytes in the structure.

Id: {e1e8410b-c8c4-4022-93ba-b2145ed6d134} is the **objectGUID** value associated with ABSUser5 in the sample Active Directory data.

NumberOfAttributes: 0x0006 specifies the number of **ABF_CONTACT_ATTRIBUTE** structures that follow this structure.

3.1.5.2 ABF_CONTACT_ATTRIBUTE telephoneNumber

```
00000C10:             0A 35 35 35 2D 37 38 39 2D 36 36 36     .555-789-666
00000C20: 36 00                                           6.
```

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
"555-789-6666"																Id (0x0A)															

Id: 0x0A corresponds to the **telephoneNumber ABF_ATTRIBUTE** structure.

Data: "555-789-6666" is a null-terminated string that is the value of the **telephoneNumber** attribute for this contact.

3.1.5.3 ABF_CONTACT_ATTRIBUTE telephoneNumber, normalized

```
00000C20:       0A 74 65 6C 3A 2B 35 35 35 37 38 39 36 36   .tel:+55578966
00000C30: 36 36 00                                        66.
```

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
"tel:+5557896666 "																Id (0x0A)															

Id: 0x0A corresponds to the **telephoneNumber ABF_ATTRIBUTE** structure.

Data: "tel:+5557896666 " is a null-terminated string that is the normalized value of the **telephoneNumber** attribute for this contact.

3.1.5.4 ABF_CONTACT_ATTRIBUTE msRTCSIP-PrimaryUserAddress

```
00000C30:          09 73 69 70 3A 41 42 53 55 73 65 72 35    .sip:ABSUser5
00000C40: 40 75 72 74 65 73 74 2E 72 74 6D 70 2E 73 65 6C @urtest.rtmp.sel
00000C50: 66 68 6F 73 74 2E 63 6F 72 70 2E 70 72 6F 73 65 fhost.corp.prose
00000C60: 77 61 72 65 2E 63 6F 6D 00                       ware.com.
```

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
"sip:ABSUser5@urtest.rtmp.selfhost.corp.proseware.com"																Id (0x09)															

Id: 0x09 corresponds to the **msRTCSIP-PrimaryUserAddress ABF_ATTRIBUTE** structure.

Data: "sip:ABSUser5@urtest.rtmp.selfhost.corp.proseware.com" is a null-terminated string that is the value of the **msRTCSIP-PrimaryUserAddress** attribute for this contact.

3.1.5.5 ABF_CONTACT_ATTRIBUTE displayName

```
00000C60:                          04 41 42 53 55 73 65          .ABSUse
00000C70: 72 35 5F 64 69 73 70 6C 61 79 6E 61 6D 65 00    r5_displayname.
```

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
"ABSUser5_displayname"																Id (0x04)															

Id: 0x04 corresponds to the **displayName ABF_ATTRIBUTE** structure.

Data: "ABSUser5_displayname" is a null-terminated string that is the value of the **displayName** attribute for this contact.

3.1.5.6 ABF_CONTACT_ATTRIBUTE sn

```
00000C70:                                             03             .
00000C80: 41 42 53 55 73 65 72 35 5F 6C 61 73 74 6E 61 6D ABSUser5_lastnam
00000C90: 65 00                                          e.
```

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
"ABSUser5_lastname"																Id (0x03)															

Id: 0x03 corresponds to the **sn ABF_ATTRIBUTE** structure.

Data: "ABSUser5_lastname" is a null-terminated string that is the value of the **sn** attribute for this contact.

3.1.5.7 ABF_CONTACT_ATTRIBUTE givenName

```
00000C90:       02 41 42 53 55 73 65 72 35 5F 66 69 72 73   .ABSUser5_firs
00000CA0: 74 6E 61 6D 65 00                                 tname.
```

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
"ABSUser5_firstname"																Id (0x02)															

Id: 0x02 corresponds to the **givenName** ABF_ATTRIBUTE structure.

Data: "ABSUser5_firstname" is a null-terminated string that is the value of the **givenName** attribute for this contact.

3.1.6 ABSUser2 Contact

3.1.6.1 ABF_CONTACT

```
00000CA0:                   A6 00 1B 25 7C 47 42 DB F6 4E       ...%|GB..N
00000CB0: 9C 69 FA BB EC C6 7F 31 00 00 06 00             .i.....1....
```

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
Id (0x251B)																Length (0x00A6)															
0xDB42477C																															
0x699C4EF6																															
0xC6ECBBFA																															
NumberOfAttributes (0x0000)																0x317F															
																NumberOfDeletedAttributes (0x0006)															

Length: 0x00A6 specifies the length of the remaining bytes in the structure.

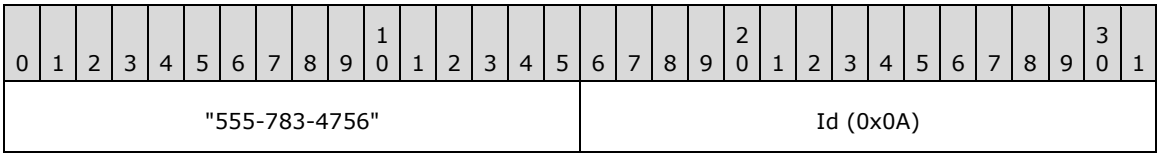
Id: {477c251b-db42-4ef6-9c69-fabbecc67f31} is the **objectGUID** value associated with ABSUser2 in the sample Active Directory data.

NumberOfAttributes: 0x0000 specifies that this contact was deleted. Because the **Length** field specifies a length greater than 18, the next field is also present.

NumberOfDeletedAttributes: 0x0006 specifies the number of **ABF_CONTACT_ATTRIBUTE** structures that follow this structure.

3.1.6.2 ABF_CONTACT_ATTRIBUTE telephoneNumber

```
00000CB0:                                     0A 35 35 35            .555
00000CC0: 2D 37 38 33 2D 34 37 35 36 00                  -783-4756.
```

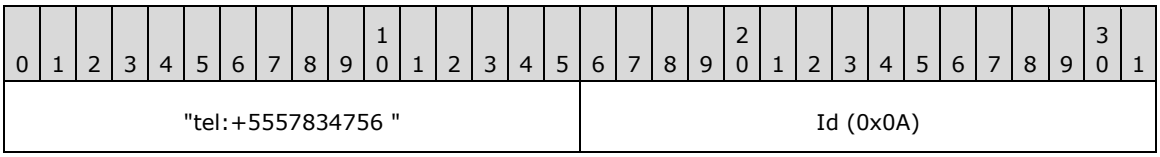


Id: 0x0A corresponds to the **telephoneNumber ABF_ATTRIBUTE** structure.

Data: "555-783-4756" is a null-terminated string that is the value of the **telephoneNumber** attribute for this contact.

3.1.6.3 ABF_CONTACT_ATTRIBUTE telephoneNumber, normalized

```
00000CC0:                                 0A 74 65 6C 3A 2B           .tel:+
00000CD0: 35 35 35 37 38 33 34 37 35 36 00               5557834756 .
```



Id: 0x0A corresponds to the **telephoneNumber ABF_ATTRIBUTE** structure.

Data: "tel:+5557834756 " is a null-terminated string that is the value of the "telephoneNumber" attribute for this contact.

3.1.6.4 ABF_CONTACT_ATTRIBUTE msRTCSIP-PrimaryUserAddress

```
00000CD0:                                     09 73 69 70 3A           .sip:
00000CE0: 41 42 53 55 73 65 72 32 40 75 72 74 65 73 74 2E ABSUser2@urtest.
00000CF0: 72 74 6D 70 2E 73 65 6C 66 68 6F 73 74 2E 63 6F rtmp.selfhost.co
00000D00: 72 70 2E 70 72 6F 73 65 77 61 72 65 2E 63 6F 6D rp.proseware.com
00000D10: 00                                             .
```

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
"sip:ABSUser2@urtest.rtmp.selfhost.corp.proseware.com"																Id (0x09)															

Id: 0x09 corresponds to the **msRTCSIP-PrimaryUserAddress ABF_ATTRIBUTE** structure.

Data: "sip:ABSUser2@urtest.rtmp.selfhost.corp.proseware.com" is a null-terminated string that is the value of the **msRTCSIP-PrimaryUserAddress** attribute for this contact.

3.1.6.5 ABF_CONTACT_ATTRIBUTE displayName

```
00000D10:    04 41 42 53 55 73 65 72 32 5F 64 69 73 70 6C  .ABSUser2_displ
00000D20: 61 79 6E 61 6D 65 00                            ayname.
```

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
"ABSUser2_displayname"																Id (0x04)															

Id: 0x04 corresponds to the **displayName ABF_ATTRIBUTE** structure.

Data: "ABSUser2_displayname" is a null-terminated string that is the value of the **displayName** attribute for this contact.

3.1.6.6 ABF_CONTACT_ATTRIBUTE sn

```
00000D20:                    03 41 42 53 55 73 65 72 32        .ABSUser2
00000D30: 5F 6C 61 73 74 6E 61 6D 65 00                    lastname.
```

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
"ABSUser2_lastname"																Id (0x03)															

Id: 0x03 corresponds to the **sn ABF_ATTRIBUTE** structure.

Data: "ABSUser2_lastname" is a null-terminated string that is the value of the **sn** attribute for this contact.

3.1.6.7 ABF_CONTACT_ATTRIBUTE givenName

```
00000D30:                              02 41 42 53 55 73           .ABSUs
00000D40: 65 72 32 5F 66 69 72 73 74 6E 61 6D 65 00       er2 firstname.
```

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
"ABSUser2_firstname"																Id (0x02)															

Id: 0x02 corresponds to the **givenName ABF_ATTRIBUTE** structure.

Data: "ABSUser2_firstname" is a null-terminated string that is the value of the **givenName** attribute for this contact.

3.1.7 ABF_CONTACT

This section provides an example of the address book file **ABF_CONTACT** structure specified in section 2.2.10.

```
00000D40:                                           12 00              ..
00000D50: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
00000D60: 00 00                                        ....
```

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
Id (0x0000)																Length (0x0012)															
0x00000000																															
0x00000000																															
0x00000000																															
NumberOfAttributes (0x0000)																0x0000															

Length: 0x0012 specifies the length of the remaining bytes in the structure.

Id: {00000000-0000-0000-0000-000000000000} is a GUID with a value of 0, which indicates the end of the **ABF_CONTACT** array.

NumberOfAttributes: 0x0000 specifies that this contact was deleted, unless the GUID in the **Id** field has a value of 0, in which case this structure indicates the end of the ABF_CONTACT array.

3.1.8 ABF_DELTA_TRAILER

This section provides an example of the address book file **ABF_DELTA_TRAILER** structure specified in section 2.2.13.

```
00000D60:       DB DF D2 DD 02 00 00 00 01 00 00 00         ............
```

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
BaseFileHash (0xDDD2)																Hash (0xDFDB)															
NumberOfContacts (0x00000002)																															
NumberOfDeletedContacts (0x00000001)																															

Hash: 0xDFDB is the hash of the contents of the file. The cryptographic hash function that is used can be anything, as long as it is the same function for all files.

BaseFileHash: 0xDDD2 is the hash of the contents of the base file that was used to calculate this delta file. The hash function that is used can be anything, as long as it is the same function for all files.

NumberOfContacts: 0x00000002 specifies the number of **ABF_CONTACT** structures in the file that have a nonzero value in the **NumberOfAttributes** field.

NumberOfDeletedContacts: 0x00000001 specifies the number of **ABF_CONTACT** structures in the file that have a zero value in the **NumberOfAttributes** field.

3.1.9 ABF_TRAILER_LENGTH

This section provides an example of the address book file **ABF_TRAILER_LENGTH** structure specified in section 2.2.14.

```
00000D60:                                       0C 00              ..
00000D70: 00 00                                       ..
```

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
LengthOfTrailer (0x0000000C)																															

LengthOfTrailer: 0x0000000C length of the **ABF_DELTA_TRAILER** structure in this file. This allows additional information to be placed in the file after the **ABF_DELTA_TRAILER** structure and before the **ABF_TRAILER_LENGTH** structure without affecting older clients.

3.2 Compressed Address Book File

Using the compressed form of the D-0A10-0A11.lsabs file seen in Address Book File Example, the hexadecimal dump of the compressed file is as follows:

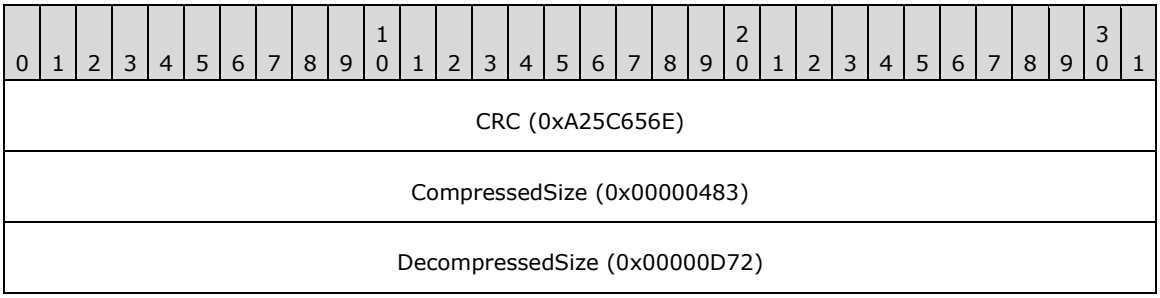
```
00000000: 6e 65 5c a2 83 04 00 00 72 0d 00 00 20 00 00 00 ne\.....r... ...
00000010: 16 c1 4b b5 08 90 c7 47 b9 bd f3 bb 1a 0a b6 eb ..K....G........
00000020: 10 0a 11 0a 14 00 14 00 01 00 07 00 2f 67 5b 08 ............/g[.
00000030: 00 00 2e 11 00 00 00 2a 38 38 32 5b 5c 73 28 29 .......*882[\s()
00000040: 5c 2d 5c 2e 2f 5d 2a 38 30 38 30 2e 2a 5b 58 78 \-\./]*8080.*[Xx
00000050: 5d 2b b7 00 28 5c 64 0d 00 00 00 00 00 29 0d 0a ]+..(\d......)..
00000060: 24 31 3b 70 68 6f 6e 65 2d 63 6f 6e 74 65 78 74 $1;phone-context
00000070: 3d 70 72 6F 73 65 77 61 72 65 2e 63 6f 60 00 00 =proseware.co`..
00000080: 10 6d 0d 0a af 02 9d 0d 0a 6e 75 6c 6c 0d 0a 5c .m.......null..\
00000090: 28 28 5c 2b 5c 73 2a 31 29 3f 5c 29 30 01 21 01 ((\+\s*1)?\)0.!.
000000a0: 5d 2a 5c 28 3f 0c 01 f0 a3 2c 03 29 d8 00 5c 29 ]*\(?....,.)..\)
```

```
000000b0: 3f ff 03 69 01 9f 04 f8 79 01 8f 05 04 bf 00 f2 ?..i....y.......
000000c0: 0d 0a 2b 31 24 32 24 33 24 34 3b c9 05 24 35 0d ..+1$2$3$4;..$5.
000000d0: 0a 6e 04 5f 04 2e 45 58 1c 21 ff 70 54 28 00 97 .n._..EX.!.pT(..
000000e0: 09 f4 a7 03 48 5b 78 58 5d af 03 2f 0b 99 03 97 ....H[xX]../....
000000f0: 03 9f 01 f0 af 03 18 a8 00 7f 0a f2 9f 06 18 37 ...............7
00000100: 30 6f 05 f8 37 30 24 33 6b 05 33 0d 0a 02 01 9f 0o..70$3k.3.....
00000110: 0f 06 17 02 f6 34 32 30 e2 3f 7f 35 18 02 48 02 .....420.?.5..H.
00000120: 2e 02 df 11 0d 3f 02 f9 2f 02 12 8f 01 71 0d 0a .....?../....q..
00000130: 79 02 57 0c 9f 19 9f 07 4d 09 3c 09 b5 04 97 0e y.W.....M.<.....
00000140: cf 04 13 70 03 30 31 31 10 01 2b 29 28 d7 04 71 ...p.011..+)(..q
00000150: 06 2b 29 29 3f 40 00 c0 c0 9f 00 1f 9b 8f 0c 7b .+))?@.........{
00000160: 31 2c 31 35 7d 46 13 b0 07 33 24 35 24 37 24 39 1,15}F...3$5$7$9
00000170: 24 31 31 24 31 33 24 31 48 08 37 24 31 39 24 32 $11$13$1H.7$19$2
00000180: 80 f9 06 7f 31 74 08 7f 08 ff ba ef 07 0b fa 14 ....1t..........
00000190: 97 07 fb 97 00 0d 67 04 ff 25 0d 0a 45 31 36 f8 ......g..%..E16.
000001a0: 1f f4 2e 2b 63 01 5f 0b 9b 7f 14 ff 3b 7f 08 b7 ...+c._.....;...
000001b0: 7f 14 1f 0d 00 0e 20 47 00 00 10 6d 61 6e 61 67 ...... G...manag
000001c0: 65 40 00 00 00 72 00 10 00 13 00 01 08 01 0f 67 e@...r.........g
000001d0: 72 6f 75 70 54 79 70 65 00 15 00 12 00 02 38 48 roupType......8H
000001e0: 70 72 6f 78 79 41 0d 20 08 00 64 64 72 65 73 73 proxyA. ..ddress
000001f0: 65 73 00 0b 00 11 c1 01 0e 6d 61 69 6c 28 02 10 es.......mail(..
00000200: 00 00 20 00 0d 69 70 50 e9 29 98 01 0f 79 00 82 .. ..ipP.)...y..
00000210: 00 35 00 0c 6f 74 68 65 72 54 65 6c 65 a2 2a 40 .5..otherTele.*@
00000220: 02 0e b1 00 0b b2 00 4d 6f 62 69 6c 65 00 0d 08 .......Mobile...
00000230: 00 00 28 02 0b 6d 73 00 15 84 44 50 28 00 0c 11 ..(..ms...DP(...
00000240: 01 0a 12 01 48 6f 6d 65 83 02 10 78 03 00 28 00 ....Home...x..(.
00000250: 0a 68 8e 00 16 00 0a b9 01 09 74 ed 02 4e 75 6d .h........t..Num
00000260: 62 00 06 22 00 22 00 00 00 09 00 00 08 02 08 6d b.."."........m
00000270: 73 52 54 43 53 49 50 2d 50 72 69 6d 61 72 79 55 sRTCSIP-PrimaryU
00000280: 73 65 72 ec 05 00 21 00 a0 4a 00 0a 00 00 00 00 ser...!..J......
00000290: 07 70 68 79 73 69 63 61 6c 44 65 6c 69 76 65 72 .physicalDeliver
000002a0: 79 4f 66 66 69 63 65 4e 61 6d 65 40 05 07 11 01 yOfficeName@....
000002b0: 06 14 22 08 84 a8 2f 70 61 6e 79 a8 08 06 00 00 .."".../pany.....
000002c0: 04 00 05 71 07 4e 69 63 6b 6e 21 01 0c 00 05 21 ...q.Nickn!....!
000002d0: 01 04 74 69 74 d0 05 12 f8 00 00 00 a0 52 44 00 ..tit........RD.
000002e0: 02 03 64 69 73 70 6c 61 79 32 02 09 00 03 09 01 ..display2......
000002f0: 02 73 6e 10 06 02 51 00 01 67 50 03 6e e2 00 21 .sn...Q..gP.n..!
00000300: f3 52 ff 6d 73 45 78 00 00 20 00 63 68 48 69 64 .R.msEx.. .chHid
00000310: 65 46 72 6f 6d 24 05 4c 69 73 74 73 00 66 01 0a eFrom$.Lists.f..
00000320: ad 36 8c 97 5e dd 46 8d 5b 25 51 40 00 00 01 21 .6..^.F.[%Q@...!
00000330: c5 2b 28 03 11 41 42 53 69 06 31 40 75 72 74 65 .+(..ABSi.1@urte
00000340: 73 ca 34 00 0f 35 35 35 2d 35 33 33 2d 34 33 31 s.4..555-533-431
00000350: 32 00 0f 3a 00 11 9d a8 08 3a 2b 90 00 88 00 82 2..:.....:+.....
00000360: 00 0d f7 00 33 00 0d f7 00 24 33 00 0c f1 00 33 ....3....$3....3
00000370: 39 31 2d 33 30 34 32 00 0c f5 00 88 00 82 00 0b 91-0003.........
00000380: ea 01 36 40 30 a0 52 36 e3 02 0b ee 01 36 36 82 ..6@0.R6.....66.
00000390: 00 0a 50 00 20 58 01 20 33 32 32 34 00 0a ef 01 ..P. X. 0001....
000003a0: 81 00 09 73 69 70 3a 9f 05 36 72 74 6d 70 2e 73 ...sip:..6rtmp.s
000003b0: 00 0a 40 04 65 6c 66 68 6f 0a 06 72 70 2e 57 3b ..@.elfho..rp.W;
000003c0: 00 08 31 32 33 34 35 00 07 54 68 01 43 24 0c 05 ..12345..Th.C$..
000003d0: 44 65 76 65 6c 6f 70 6d 28 04 60 05 65 6e 74 20 Developm(.`.ent
000003e0: 4d 8c 15 04 d5 02 5f c4 0b d1 0c 5f 63 68 61 6e M..... .... chan
000003f0: 67 65 64 00 03 ee 00 6c 61 73 74 aa 0d 02 96 00 ged....last.....
00000400: 66 69 72 1f 08 00 80 9c 00 a4 00 0b 41 e8 e1 c4 fir.........A...
00000410: c8 22 40 93 ba b2 14 5e d6 d1 34 06 b2 06 2d 37 ."@....^..4...-7
00000420: 38 39 2d 36 00 00 b7 06 20 88 00 82 00 b7 06 0e 89-6.... .......
00000430: 00 40 6d 35 b7 06 2f 10 6d 05 35 6f 05 2e 05 35 .@m5../.m.5o...5
00000440: 2f 05 18 35 2f 05 a6 00 1b 25 7c 47 42 db f6 4e /..5/....%|GB..N
00000450: 9c 69 fa bb ec c6 7f 31 08 15 3d 05 d8 0e 37 07 .i.....1..=...7.
00000460: 08 ab 5a 35 3f 05 23 33 82 00 3f 05 32 3f 05 bf ..Z5?.#3..?.2?..
00000470: 18 32 3f 05 32 3f 05 18 32 3f 05 12 ee 65 07 00 .2?.2?..2?...e..
00000480: 00 db df d2 dd 01 18 01 00 00 00 0c 00 00 00    ..............
```

3.3 COMPRESSED_BLOCK_HEADER

Each **COMPRESSED_BLOCK** structure in a compressed address book file begins with a **COMPRESSED_BLOCK_HEADER** structure that gives the size of the remaining bytes in the **COMPRESS_BLOCK** structure.

```
00000000: 6e 65 5c a2 83 04 00 00 72 0d 00 00            ne\.....r...
```



CRC: 0xA25C656E is the CRC of the decompressed **Data** bytes in the block. The algorithm that is used to compute this value is described in the next section.

CompressedSize: 0x00000483 is the length, in bytes, of the **Data** field that follows the **COMPRESSED_BLOCK_HEADER**.

DecompressedSize: 0x00000D72 is the expected length, in bytes, of the decompressed data.

4 Security Considerations

4.1 Security Considerations for Implementers

None.

4.2 Index of Security Fields

None.

5 Appendix A: Compression Format

The information in this section describes the format of compressed Address Book Files with examples and pseudo code to decompress a compressed file.

5.1 32-Bit CRC Algorithm

This section uses pseudo code to explain how to generate the 32-bit CRC value stored in the **CRC** field of the **COMPRESSED_BLOCK_HEADER** structure.

```
Define CalculateCRC32 as Function With Parameters (
    buffer as unsigned char []
    length as unsigned integer
    )
Begin
    Define crc as unsigned integer
    Define index as integer

    crc = 0xFFFFFFFF
    index := 0
    While (index < length)
    Begin
        crc := crc32_table [(crc AND 0xFF) XOR buffer [index]] ^ (crc / 256)
        index := index + 1
    End
    return crc XOR 0xFFFFFFFF
End

Define crc32_table as unsigned integer [256] =
Begin
    0x00000000, 0x77073096, 0xEE0E612C, 0x990951BA,
    0x076DC419, 0x706AF48F, 0xE963A535, 0x9E6495A3,
    0x0EDB8832, 0x79DCB8A4, 0xE0D5E91E, 0x97D2D988,
    0x09B64C2B, 0x7EB17CBD, 0xE7B82D07, 0x90BF1D91,
    0x1DB71064, 0x6AB020F2, 0xF3B97148, 0x84BE41DE,
    0x1ADAD47D, 0x6DDDE4EB, 0xF4D4B551, 0x83D385C7,
    0x136C9856, 0x646BA8C0, 0xFD62F97A, 0x8A65C9EC,
    0x14015C4F, 0x63066CD9, 0xFA0F3D63, 0x8D080DF5,
    0x3B6E20C8, 0x4C69105E, 0xD56041E4, 0xA2677172,
    0x3C03E4D1, 0x4B04D447, 0xD20D85FD, 0xA50AB56B,
    0x35B5A8FA, 0x42B2986C, 0xDBBBC9D6, 0xACBCF940,
    0x32D86CE3, 0x45DF5C75, 0xDCD60DCF, 0xABD13D59,
    0x26D930AC, 0x51DE003A, 0xC8D75180, 0xBFD06116,
    0x21B4F4B5, 0x56B3C423, 0xCFBA9599, 0xB8BDA50F,
    0x2802B89E, 0x5F058808, 0xC60CD9B2, 0xB10BE924,
    0x2F6F7C87, 0x58684C11, 0xC1611DAB, 0xB6662D3D,
    0x76DC4190, 0x01DB7106, 0x98D220BC, 0xEFD5102A,
    0x71B18589, 0x06B6B51F, 0x9FBFE4A5, 0xE8B8D433,
    0x7807C9A2, 0x0F00F934, 0x9609A88E, 0xE10E9818,
    0x7F6A0DBB, 0x086D3D2D, 0x91646C97, 0xE6635C01,
    0x6B6B51F4, 0x1C6C6162, 0x856530D8, 0xF262004E,
    0x6C0695ED, 0x1B01A57B, 0x8208F4C1, 0xF50FC457,
    0x65B0D9C6, 0x12B7E950, 0x8BBEB8EA, 0xFCB9887C,
    0x62DD1DDF, 0x15DA2D49, 0x8CD37CF3, 0xFBD44C65,
    0x4DB26158, 0x3AB551CE, 0xA3BC0074, 0xD4BB30E2,
    0x4ADFA541, 0x3DD895D7, 0xA4D1C46D, 0xD3D6F4FB,
    0x4369E96A, 0x346ED9FC, 0xAD678846, 0xDA60B8D0,
    0x44042D73, 0x33031DE5, 0xAA0A4C5F, 0xDD0D7CC9,
    0x5005713C, 0x270241AA, 0xBE0B1010, 0xC90C2086,
    0x5768B525, 0x206F85B3, 0xB966D409, 0xCE61E49F,
    0x5EDEF90E, 0x29D9C998, 0xB0D09822, 0xC7D7A8B4,
    0x59B33D17, 0x2EB40D81, 0xB7BD5C3B, 0xC0BA6CAD,
    0xEDB88320, 0x9ABFB3B6, 0x03B6E20C, 0x74B1D29A,
    0xEAD54739, 0x9DD277AF, 0x04DB2615, 0x73DC1683,
    0xE3630B12, 0x94643B84, 0x0D6D6A3E, 0x7A6A5AA8,
    0xE40ECF0B, 0x9309FF9D, 0x0A00AE27, 0x7D079EB1,
```

```
    0xF00F9344, 0x8708A3D2, 0x1E01F268, 0x6906C2FE,
    0xF762575D, 0x806567CB, 0x196C3671, 0x6E6B06E7,
    0xFED41B76, 0x89D32BE0, 0x10DA7A5A, 0x67DD4ACC,
    0xF9B9DF6F, 0x8EBEEFF9, 0x17B7BE43, 0x60B08ED5,
    0xD6D6A3E8, 0xA1D1937E, 0x38D8C2C4, 0x4FDFF252,
    0xD1BB67F1, 0xA6BC5767, 0x3FB506DD, 0x48B2364B,
    0xD80D2BDA, 0xAF0A1B4C, 0x36034AF6, 0x41047A60,
    0xDF60EFC3, 0xA867DF55, 0x316E8EEF, 0x4669BE79,
    0xCB61B38C, 0xBC66831A, 0x256FD2A0, 0x5268E236,
    0xCC0C7795, 0xBB0B4703, 0x220216B9, 0x5505262F,
    0xC5BA3BBE, 0xB2BD0B28, 0x2BB45A92, 0x5CB36A04,
    0xC2D7FFA7, 0xB5D0CF31, 0x2CD99E8B, 0x5BDEAE1D,
    0x9B64C2B0, 0xEC63F226, 0x756AA39C, 0x026D930A,
    0x9C0906A9, 0xEB0E363F, 0x72076785, 0x05005713,
    0x95BF4A82, 0xE2B87A14, 0x7BB12BAE, 0x0CB61B38,
    0x92D28E9B, 0xE5D5BE0D, 0x7CDCEFB7, 0x0BDBDF21,
    0x86D3D2D4, 0xF1D4E242, 0x68DDB3F8, 0x1FDA836E,
    0x81BE16CD, 0xF6B9265B, 0x6FB077E1, 0x18B74777,
    0x88085AE6, 0xFF0F6A70, 0x66063BCA, 0x11010B5C,
    0x8F659EFF, 0xF862AE69, 0x616BFFD3, 0x166CCF45,
    0xA00AE278, 0xD70DD2EE, 0x4E048354, 0x3903B3C2,
    0xA7672661, 0xD06016F7, 0x4969474D, 0x3E6E77DB,
    0xAED16A4A, 0xD9D65ADC, 0x40DF0B66, 0x37D83BF0,
    0xA9BCAE53, 0xDEBB9EC5, 0x47B2CF7F, 0x30B5FFE9,
    0xBDBDF21C, 0xCABAC28A, 0x53B39330, 0x24B4A3A6,
    0xBAD03605, 0xCDD70693, 0x54DE5729, 0x23D967BF,
    0xB3667A2E, 0xC4614AB8, 0x5D681B02, 0x2A6F2B94,
    0xB40BBE37, 0xC30C8EA1, 0x5A05DF1B, 0x2D02EF8D
End
```

5.2 Compressed Data Format

The compression technique used to compress the output files belongs to the LZ77 family, as described in [UASDC], and is based on a simple idea: compression is achieved by substitution of substrings with pointers to previous occurrence of exactly the same substring in form of **(offset, length)** where **offset** is offset to the beginning of matching substring and **length** is its length

At the highest level, each block of compressed data consists of one or more token groups. All but the last token group has 32 tokens. Each token group starts out with a 32-bit unsigned integer, where each bit indicates the type of one token. Bit 0 gives the type of the first token, Bit 1 the type of the second token, and so on. The data for each token is serialized after the 32-bit unsigned integer. If a bit in the unsigned integer is 0, then the corresponding token is a literal byte, which is copied directly from the input buffer to the output buffer. If the bit is 1, then the corresponding token is a reference to a sequence of 3 or more bytes from a previous position in the output buffer. The length and offset of the data to be copied are encoded in the data for the token. This encoding occupies a minimum of 2 bytes and a maximum of 6 bytes, depending on the length of the data being copied.

5.3 Run Encoding

As described in the previous section, when a run of characters is encoded, a length and offset are stored in a minimum of 2 bytes and a maximum of 6. This encoding works by storing the offset in the high 13 bits of the first two bytes and the length in the low 3 bits. Because the compression algorithm does not generate runs shorter than 3 bytes, the length is biased by -3.

The first 2 bytes are treated as an unsigned 16-bit integer, with the high 13 bits being the offset and the low 3 bits being the length. The offset has been biased by -1, so in order for the number to be useful, 1 must be added back to it. The offset is a backwards offset from the current position in the output buffer. So after adjusting for the -1 bias, an offset of zero in the high 13 bits would be 1, and that value (1) would then be subtracted from the current position in the output buffer to get the location of the previous character in the buffer that starts the run.

The length field is biased by -3, because no runs of length less than 3 can be emitted by the compression algorithm. So a value of 0 in the low 3 bits is really a length of 3. This bias makes it possible to encode lengths of 3 to 10 (0 through 7) in 3 bits. Because the length will often be greater than 10 bits, there is an additional mechanism: If all 3 bits are set (0x7), then the next byte is examined. The first time, the low 4 bits of that byte are extracted as the length, and the location of the byte is stored. This 4-bit length is biased by -10 (-(3+7)), so it is now possible to represent lengths of 10 through 25 using the three-byte notation. In fact, only two and a half bytes are available, because each run of length 10 to 25 uses either the low or high four bits of the byte. Again, to accommodate lengths greater than 25, there is another escape mechanism: If all 4 bits are 1 (0xF), then the next byte is also examined. If the next byte is not equal to 255 (0xFF), then that is the length, biased by -25 (-(3+7+15)), so it is possible to encode lengths of 25 through 279. Finally, if the fourth byte is 255 (0xFF), then the next two bytes contain the length, biased by -3. The following section contains pseudo code to demonstrate this encoding algorithm. The decompression pseudo code in the section after that shows how to decode the encoded offset and length.

5.4 Pseudo Code to Encode Offset and Length into a Token

While this code is necessary to generate a well formed compressed file, the bulk of the complexity in any LZ77 compression algorithm is the logic to find duplicate runs of bytes in the input. That logic is beyond the scope of this specification.

```
Define NibbleIndex as integer := -1

Define EncodeOffsetLength as Function With Parameters (
    offset as unsigned integer
    length as unsigned integer
    outputBuffer as unsigned char array
    outputIndex as unsigned integer
    )
Begin
    Define nibbleValue as unsigned integer

    offset := offset - 1
    length := length - 3

    offset := offset times 8
    If (length < 7)
    Begin
        offset := offset OR length
        outputBuffer [outputIndex] = offset
        outputBuffer [outputIndex+1] = offset / 256
        return outputIndex + 2
    End

    offset := offset OR 0x7
    length -= 7;

    outputBuffer [outputIndex] = offset
    outputBuffer [outputIndex+1] = offset / 256
    outputIndex := outputIndex + 2

    nibbleValue = length
    If (nibbleValue >= 15)
    Begin
        nibbleValue = 15
    End

    If (NibbleIndex equals -1)
    Begin
        NibbleIndex := outputIndex
        outputBuffer [NibbleIndex] := nibbleValue
        outputIndex := outputIndex + 1
    End
    Else
```

```
        Begin
            outputBuffer [NibbleIndex] := outputBuffer [NibbleIndex] OR (nibbleValue times 16)
            NibbleIndex := -1
        End
        If (nibbleValue equals 15)
        Begin
            length := length - 15
            If (length >= 255)
            Begin
                length := length + 7 + 15
                outputBuffer [outputIndex] := 255
                outputBuffer [outputIndex + 1] := length
                outputBuffer [outputIndex + 2] := length / 256
                outputIndex := outputIndex + 3
            End
            Else
            Begin
                outputBuffer [outputIndex] := length
                outputIndex := outputIndex + 1
            End
        End

        return outputIndex
    End
```

5.5 Pseudo Code to Decompress an Address Book File

Decompressing a compressed file consists of the following steps:

1. Looping over the input file.

2. Reading a **COMPRESSED_BLOCK_HEADER**.

3. Reading from that header the **Data** portion of the **COMPRESSED_BLOCK** based on the **CompressedSize** field in the **COMPRESSED_BLOCK_HEADER** structure.

4. Allocating an output buffer based on the **DecompressedSize** field in the **COMPRESSED_BLOCK_HEADER** structure and decompressing the input buffer into the output buffer according to the algorithm described in earlier in this section.

The following pseudo code implements this procedure. The pseudo code assumes that file handles of some sort have been opened to the input file to be decompressed and to the decompressed file that is being created. It also assumes the existences of **ReadFile** and **WriteFile** primitives to read and write bytes to and from those files handles. In the interest of brevity, there is no error checking logic in the following pseudo code, and the pseudo code assumes a well-formed input file.

5.5.1 Function to Decompress a File

```
    Define DecompressFile as Function With Parameters (
        inputFile as FileHandle,
        outputFile as FileHandle
        )
    Begin
        Define blockHeader as structure
        Begin
            Define crc32 as unsigned integer
            Define compressedSize as unsigned integer
            Define decompressedSize as unsigned integer
        End
        Define bufferSize as unsigned integer := 65536
        Define inputBuffer as unsigned char [bufferSize]
        Define outputBuffer as unsigned char [bufferSize]
```

```
    Define bytesRead as unsigned integer

    While (ReadFile (inputFile, blockHeader, sizeof (blockHeader)) not equal 0)
    Begin
        ReadFile (inputFile, inputBuffer, blockHeader.compressedSize);
        If (blockHeader.compressedSize < blockHeader.decompressedSize)
        Begin
            DecompressBuffer (inputBuffer, outputBuffer, blockHeader.decompressedSize);
            WriteFile (outputFile, outputBuffer, blockHeader.decompressedSize);
        End
        Else
        Begin
            // Block is not compressed, just write to output file
            WriteFile (outputFile, inputBuffer, blockHeader.compressedSize);
        End
    End
End
```

5.5.2 Function to Decompress the Bytes in a Block

The following pseudo code shows how to decompress the bytes in the **Data** field of a **COMPRESSED_BLOCK** structure. The Print statements are for debugging purposes and are not necessary, nor are the tokenGroup and tokenIndex variables, which are there to support the Print statements.

```
Define DecompressBuffer as Function With Parameters (
    inputBuffer as unsigned char []
    outputBuffer as unsigned char []
    outputSize as unsigned integer
    )
Begin
    Define inputIndex as unsigned integer
    Define outputIndex as unsigned integer
    Define tokenType as unsigned integer
    Define tokenTypeMask as unsigned integer
    Define nibbleIndex as integer
    Define offset as integer
    Define length as unsigned integer
    Define tokenGroup as unsigned integer
    Define tokenIndex as unsigned integer

    inputIndex := 0
    outputIndex := 0
    tokenTypeMask := 0
    nibbleIndex := -1
    tokenGroup := 1

    While (outputIndex < outputSize)
    Begin
        If (tokenTypeMask < 2)
        Begin
            tokenType := inputBuffer [inputIndex] |
                         inputBuffer [inputIndex + 1] times 256 |
                         inputBuffer [inputIndex + 2] times 256 times 256 |
                         inputBuffer [inputIndex + 3] times 256 times 256 times 256
            inputIndex := inputIndex + 4
            tokenTypeMask := 0x80000000
            Print "Token Group [" + tokenGroup + "]: tokenType: " + tokenType + NewLine
            tokenGroup := tokenGroup + 1
        End
        Else
        Begin
            tokenTypeMask := tokenTypeMask / 2
        End
```

```
        Print "  Token [" + tokenIndex + "]: "
        tokenIndex := tokenIndex + 1
        If ((tokenType & tokenTypeMask) equal 0)
        Begin
            Print "  LITERAL: " + inputBuffer [inputIndex] +
                  " copied to " + outputIndex + NewLine
            outputBuffer [outputIndex] := inputBuffer [inputIndex]
            inputIndex := inputIndex + 1
            outputIndex := outputIndex + 1
        End
        Else
        Begin
            offset := inputBuffer [inputIndex] | (inputBuffer [inputIndex+1] times 256)
            Print "  RUN:" + offset
            length := offset & 0x7
            offset := offset / 8
            inputIndex := inputIndex + 2

            If (length equals 7)
            Begin
                If (nibbleIndex equals -1)
                Begin
                    Print " " + inputBuffer [inputIndex] +" (low nibble)"
                    nibbleIndex := inputIndex
                    length := inputBuffer [nibbleIndex] & 0xF
                    inputIndex := inputIndex + 1
                End
                Else
                Begin
                    Print " " + inputBuffer [nibbleIndex] +" (high nibble)"
                    length := inputBuffer [nibbleIndex] / 16
                    nibbleIndex := -1
                End

                If (length equals 15)
                Begin
                    Print " " + inputBuffer [inputIndex]
                    length := inputBuffer [inputIndex]
                    inputIndex := inputIndex + 1
                    If (length equals 255)
                    Begin
                        length := inputBuffer [inputIndex] |
                                  inputBuffer [inputIndex + 1] times 256
                        Print " " + length
                        inputIndex := inputIndex + 2
                        length := length - 15 - 7
                    End

                    length := length + 15
                End

                length := length + 7
            End
            length := length + 3

            Print " - length: " + length + "  offset: " + offset +
                  "  Data at " + (outputIndex - offset - 1) +
                  ": copied to " + outputIndex + NewLine
            While (length not equals 0)
            Begin
                outputBuffer [outputIndex] := outputBuffer [outputIndex - offset - 1]
                Print " " + outputBuffer [outputIndex]
                outputIndex := outputIndex + 1
                length := length - 1
            End
            Print NewLine
        End
    End
```

```
End
```

6 Appendix B: Hash Function

The ABF_FULL_TRAILER and ABF_DELTA_TRAILER structures contain hash fields. The cryptographic hash function used to compute these fields is implementation-specific, with the requirement that the same function be used in computing each **hash code** for each Address Book Server file in a deployment and that the hash value have a high probability of being unique for each file<13>. The following pseudo code is an example of a cryptographic hash function.

```
Define HashContacts as Function With Parameters (
    contacts as ABF_CONTACT []
    numberOfContacts as unsigned integer
    )
Begin
    Define contactIndex as unsigned integer
    Define hashValue as unsigned integer

    hashValue = 0
    For (contactIndex=0; contactIndex<numberOfContacts; contactIndex++)
    Begin
    hashValue = hashValue + HashContact (contacts [contactIndex])
    End

    //
    // Return 16 bit hash value
    //
    Return (hashValue & 0xFFFF) XOR (hashValue >> 16)
End

Define HashContact as Function With Parameters (
    contact as ABF CONTACT
Begin
    Define hashValue as unsigned integer
    Define attributeIndex as unsigned integer

    hashValue = HashBytes (contact.Id, SizeOf (contact.Id))
    For (attributeIndex=0; attributeIndex<contact.NumberOfAttributes; attributeIndex++)
    Begin
    hashValue = hashValue + HashContactAttribute (contact.Attributes [attributeIndex])
    End

    Return hashValue
End

Define HashContactAttribute as Function With Parameters (
    contactAttribute as ABF CONTACT ATTRIBUTE
    )
Begin
    // contactAttribute.Length may or may not be a computed field
    // depending upon the type of attribute.
    Return HashBytes (contactAttribute.Data, contactAttribute.Length)
End

Define HashBytes as Function With Parameters (
    buffer as unsigned char []
    bufferLength as unsigned integer
    )
Begin
    Define hashValue as unsigned integer
    Define bufferIndex as unsigned integer
    Define c as unsigned char

    hashValue = 0
    For (bufferIndex=0; bufferIndex<bufferLength; bufferIndex++)
    Begin
        c = buffer [bufferIndex]
    hashValue = hashValue + (c << 1) + (c >> 1) + c
    End
```

```
        Return hashValue
    End
```

7 Appendix C: Active Directory Scanning Algorithm

The primary function of the Address Book Server is to scan Active Directory for users, contacts, and groups and determine which objects to include in the Address Book Server files and, for each object included, which attributes to include. The following pseudo-code example illustrates how a scanning process works.

```
// This is a representation of the ABF ATTRIBUTES Structure
//
extern Dictionary<String, ABF ATTRIBUTE> attributeTable;
adObjects = select from Active Directory where ObjectType == Group Or
                                  ObjectType == User Or
                                  ObjectType == Contact
foreach (adObject in adObjects)
{
  hideObject = false;
  includeAttributeSeen = false;
  requiredAttributeSeen = false;
  foreach (adAttribute in adObject.Atttributes)
  {
    if (attributeTable [adAttribute.Name].Flags &
          ABF_ATTRIBUTE_FLAGS_EXCLUDE)
    {
      // Exclude attribute seen, so bail now.
      // Not going to take this adObject.
      hideObject = true;
      break;
    }
    if (attributeTable [adAttribute.Name].Flags &
          ABF_ATTRIBUTE_FLAGS_INCLUDE)
    {
      // Include attribute seen, so assume
      // this adObject will be taken
      includeAttributeSeen = true;
    }
    if (attributeTable [adAttribute.Name].Flags &
          ABF_ATTRIBUTE_FLAGS_REQUIRED)
    {
      // Required attribute seen, so assume
      // this adObject will not be taken
      requiredAttributeSeen = true;
    }
  }

  If (attributeTable.HasRequiredAttributes && !requiredAttributeSeen)
  {
      // There was at least one ABF_ATTRIBUTE_FLAGS_REQUIRED attribute
      // defined in the attributeTable, but no required
      // attributes on this adObject, so hide it.
      hideObject = true;
  }
  else
  If (attributeTable.HasIncludeAttributes && !includeAttributeSeen)
  {
      // There was at least one ABF_ATTRIBUTE_FLAGS_INCLUDE attribute
      // defined in the attributeTable, but no include
      // attributes on this adObject, so hide it.
      hideObject = true;
  }

  if (hideObject)
  {
      // Not interested in this adObject so on to the next one.
      continue;
  }

  //
```

```
      // This adObject should be in address book files so process it
      //
    }
```

8 Appendix D: 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 Communications Server 2007
- Microsoft Office Communications Server 2007 R2
- Microsoft Office Communicator 2007
- Microsoft Office Communicator 2007 R2
- Microsoft Lync Server 2010
- Microsoft Lync 2010
- Microsoft Lync Server 2013
- Microsoft Skype for Business (formerly Lync 2013)

- Skype for Business

- Skype for Business Server

Exceptions, if any, are noted below. If a service pack or Quick Fix Engineering (QFE) number appears with the product version, behavior changed in that service pack or QFE. The new behavior also applies to subsequent service packs of the product unless otherwise specified. If a product edition appears with the product version, behavior is different in that product edition.

Unless otherwise specified, any statement of optional behavior in this specification that is prescribed using the terms SHOULD or SHOULD NOT implies product behavior in accordance with the SHOULD or SHOULD NOT prescription. Unless otherwise specified, the term MAY implies that the product does not follow the prescription.

<1> Section 1.3: Office Communications Server 2007, Office Communicator 2007: Product behavior not supported. Product behavior added in relation to Knowledge Base Article 972403, July 2009. This hotfix applies to Office Communications Server 2007 R2 and Office Communicator 2007 R2.

<2> Section 1.3: Office Communications Server 2007, Office Communicator 2007: Product behavior not supported. Product behavior added in relation to Knowledge Base Article 972403, July 2009. This hotfix applies to Office Communications Server 2007 R2 and Office Communicator 2007 R2.

<3> Section 1.3.3: Office Communications Server 2007, Office Communicator 2007: This behavior is not supported. This behavior added in relation to Knowledge Base Article 972403, July 2009. This hotfix applies to Office Communications Server 2007 R2 and Office Communicator 2007 R2.

<4> Section 2.1.4: This value is not used by Communicator clients.

<5> Section 2.2.3: Office Communicator 2007 and Office Communicator 2007 R2 uses MaximumAttributeId+1 as the value of NumberOfAttributes if MaximumAttributeId is less than 256. Otherwise, it uses the NumberOfAttributes field. To work with the clients in Appendix B, values in the Id field of ABF_ATTRIBUTE structure must start at 1 and be contiguous.

<6> Section 2.2.4: Office Communications Server 2007, Office Communicator 2007: Product behavior not supported. Product behavior added in relation to Knowledge Base Article 972403, July 2009. This hotfix applies to Office Communications Server 2007 R2, Office Communicator 2007 R2.

<7> Section 2.2.4: Office Communicator 2007 and Office Communicator 2007 R2 uses MaximumAttributeId+1 as the value of NumberOfAttributes if MaximumAttributeId is less than 256. Otherwise, it uses the NumberOfAttributes field. To work with the clients in Appendix B, values in the Id field of ABF_ATTRIBUTE structure must start at 1 and be contiguous.

<8> Section 2.2.5: Office Communicator 2007, Office Communications Server 2007: The Microsoft .NET Framework 2.0 Regular Expression library is used. For all other products, Microsoft .NET Framework 3.5 Regular Expression library is used.

<9> Section 2.2.15: Office Communications Server 2007, Office Communicator 2007: This behavior is not supported. This behavior added in relation to Knowledge Base Article 972403, July 2009. This hotfix applies to Office Communications Server 2007 R2 and Office Communicator 2007 R2.

<10> Section 2.2.16: Office Communications Server 2007, Office Communicator 2007: This behavior is not supported. This behavior added in relation to Knowledge Base Article 972403, July 2009. This hotfix applies to Office Communications Server 2007 R2 and Office Communicator 2007 R2.

<11> Section 3.1: Office Communications Server 2007, Office Communicator 2007: Product behavior not supported. Product behavior added in relation to Knowledge Base Article 972403, July 2009; starting with the phrase "and a compact delta book file". This hotfix applies to Office Communications Server 2007 R2, Office Communicator 2007 R2.

<12> Section 3.1: Office Communications Server 2007, Office Communicator 2007: Product behavior not supported. Product behavior added in relation to Knowledge Base Article 972403, July 2009. This hotfix release applies to Office Communications Server 2007 R2, Office Communicator 2007 R2.

<13> Section 6: Office Communications Server 2007 calls the Microsoft .NET Framework GetHashCode method on the individual objects to compute the hash code. The other supported products call .NET Framework 3.5 GetHashCode method on the individual objects to compute the hash code.

9 Change Tracking

This section identifies changes that were made to this document since the last release. 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.
- The removal of a document from the documentation set.

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 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 changes were introduced. Minor editorial and formatting changes may have been made, but the technical content of the document is identical to the last released version.

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.
- 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 dochelp@microsoft.com.

Section	Tracking number (if applicable) and description	Major change (Y or N)	Change type
8 Appendix D: Product Behavior	Updated list of supported products.	Y	Content updated due to protocol revision.

10 Index

3

A