The Accounting Data Interchange Format (ADIF)

1. Status of this Memo

This document is an Internet-Draft. Internet-Drafts are working documents of the Internet Engineering Task Force (IETF), its areas, and its working groups. Note that other groups may also distribute working documents as Internet-Drafts.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as “work in progress.”

To learn the current status of any Internet-Draft, please check the ‘‘id-abstracts.txt’’ listing contained in the Internet-Drafts Shadow Directories on ds.internic.net (US East Coast), nic.nordu.net (Europe), ftp.isi.edu (US West Coast), or munnari.oz.au (Pacific Rim).

The distribution of this memo is unlimited. It is filed as <draft-ietf-roamops-actng-04.txt>, and expires September 1, 1998. Please send comments to the authors.

2. Abstract

This document proposes a standard accounting record format, the Accounting Data Interchange Format (ADIF), which is designed to compactly represent accounting data in a protocol-independent manner. As a result, ADIF may be used to represent accounting data from a wide variety of protocols.

3. Introduction

As detailed in [2], solution of the accounting problem in roaming requires a standardized accounting record format. Since there is no standards-track accounting protocol, the operational roaming services described in [1] exhibit considerable diversity in their accounting implementations.

In order to be able to cope with this diversity, it is desirable that a standardized accounting record format be protocol-independent. As a result, protocol-specific solutions such as the SNMP-oriented record format described in [10] do not offer sufficient flexibility.
This document proposes a standard accounting record format, based on MIME, described in [11], and resembling the LDAP Data Interchange Format (LDIF), described in [8].

3.1. Terminology

This document frequently uses the following terms:

Accounting
The act of collecting information on resource usage for the purpose of trend analysis, auditing, billing, or cost allocation.

Rating
The act of determining the price to be charged for use of a resource.

Billing
The act of preparing an invoice.

Auditing
The act of verifying the correctness of a procedure.

Cost Allocation
The act of allocating costs between entities. Note: cost allocation and rating are fundamentally different processes.

Interim accounting
An interim accounting packet provides a snapshot of usage during a user’s session. It is typically implemented in order to provide for partial accounting of a user’s session in the event of a device reboot or other network problem that prevents the reception of a session summary packet or session record.

Session record
A session record represents a summary of the resource consumption of a user over the entire session. Accounting gateways creating the session record may do so by processing interim accounting events.

Accounting Protocol
A protocol used to convey information collected for accounting purposes.

Intra-domain accounting
Intra-domain accounting involves the collection of information on resource usage of an entity within the administrative domain. As a result, accounting packets and session records will typically remain within the administrative boundary.

Inter-domain accounting
Inter-domain accounting involves the collection of information on resource usage of an entity that exists within another administrative domain. This typically results in
accounting packets, session records, or invoices crossing administrative boundaries.

Real-time accounting
Real-time accounting involves the transmission of accounting packets or session records within a defined time window. Time constraints are typically imposed so as to allow more effective management of credit risks.

4. Definition of the Accounting Data Interchange Format (ADIF)

ADIF is based on MIME, described in [11], and resembles the LDAP Data Interchange Format (LDIF) specified in [8]. An ADIF file consists of a header providing basic information about the records in the file, followed by a series of records separated by a separator.

The header includes the version number (1 for this document), device name/description, and collection start date and time. A default protocol type may be optionally included in the header.

Each record may consist of one or more lines, and as with MIME, described in [11], lines may be continued by putting a space or tab character on the succeeding line, allowing attributes to be of arbitrary length. Lines beginning with the "#" character are taken as comments and ignored. Accounting records have traditionally been human-readable, so as to allow them to be more easily debugged. As with LDIF, ADIF attributes to be expressed either in NVT ASCII (characters 32 through 126) or if non-printable characters are required, in base64.

ADIF includes support for encoding of RADIUS, SNMP, L2TP and TACACS+ attributes, and support for other protocols may be added as needed. The protocol type is indicated by prepending the protocol and a "//" to the attribute name, i.e. RADIUS//Acct-Session-Time.

To improve compactness, when a default protocol is indicated in the header, attributes of the default protocol do not need to include the protocol type, and attribute numbers may used instead of names. For example, if defaultProtocol: RADIUS is indicated in the ADIF header, then 46 may be used instead of RADIUS//Acct-Session-Time.

Protocols such as L2TP, defined in [12] include additional fields such as Vendor ID and flags in their Attribute Value Pairs (AVPs). To encode such AVPs, ADIF includes support for sub-attributes. These are included as additional fields, separated by a semi-colon, and are of the form <subattribute> = <value>. For example, the L2TP version number attribute (attribute 2) with the Mandatory bit set would be expressed as "L2TP//2: 1; M=1". This document includes support for VendorId, VendorType, Mandatory and Hidden sub-attributes; other sub-attributes may be added as needed. Use of the VendorId and VendorType subattribute may be used for expression of vendor-specific attributes, such as those supported in RADIUS.
4.1. ADIF Examples

Example 1: An ADIF file encoding RADIUS accounting data

version: 1
device: server3
descripton: Accounting Server 3
date: 02 Mar 1998 12:19:01 -0500
defaultProtocol: RADIUS

#NAS-IP-Address
4: 204.45.34.12
#NAS-Port
5: 12
#NAS-Port-Type
61: 2
#User-Name
1: fred@bigco.com
#Acct-Status-Type
40: 2
#Acct-Delay-Time
41: 14
#Acct-Input-Octets
42: 234732
#Acct-Output-Octets
43: 15439
#Acct-Session-Id
44: 185
#Acct-Authentic
45: 1
#Acct-Session-Time
46: 1238
#Acct-Input-Packets
47: 153
#Acct-Output-Packets
48: 148
#Acct-Terminate-Cause
49: 11
#Acct-Multi-Session-Id
50: 73
#Acct-Link-Count
51: 2

Example 2: An ADIF file with a vendor-specific attribute

version: 1
device: server3
descripton: Accounting Server 3
date: 02 Mar 1998 12:19:01 -0500
defaultProtocol: RADIUS

4: 204.45.34.12
5: 12
61: 2
4.2. Grammar

The following definition uses the ABNF specified in [7]:

```plaintext
adif-file            = header-spec *SEP 1*( SEP adif-record )
header-spec          = required-info SEP [optional-info SEP]
required-info        = device-spec SEP start-spec SEP
optional-info        = [version-spec SEP ] [description SEP] [default-protocol SEP]
device-spec          = "device:" *SP value
description          = "description:" *SPACE value
default-protocol     = "defaultProtocol:" *SPACE protocol
protocol             = "RADIUS" / "SNMP" / "TACACS+" / "L2TP"
version-spec         = "version:" *SPACE version-number
version-number       = 1*DIGIT ; version-number MUST be "1" for the
                       ; ADIF format described in this document
start-spec           = "date:" *SP datetime
datetime             = date SP time
date                 = Dd SP Mon SP YYYY
time                 = hh "." mm "." as SP zone
Dd                   = <the one or two decimal integer day of the month in
                       the range 1 to 31.>
Mon                  = "JAN" / "FEB" / "MAR" / "APR" / "MAY" / "JUN" /
                       "JUL" / "AUG" / "SEP" / "OCT" / "NOV" / "DEC"
YYYY                 = <the four decimal integer year in the range 0000 to
                       9999>
hh                   = <the two decimal integer hour of the day in the
                       range 00 to 24>
mm                   = <the two decimal integer minute of the hour in the
                       range 00 to 59>
ss                   = <the two decimal integer second of the minute in the
                       range 00 to 59>
zeone                = <A four digit, signed time zone offset, such as -0600 for
                       US Eastern Standard Time. This may be supplemented by a
                       time zone name in parentheses, e.g., "-0800 (PDT)">
adif-record          = 1*(attrval-series SEP)
attrval-series       = 1*(attrval-spec)
attrval-spec         = attr { (std-encoding / base-64-encoding) [sub-attr-encoding] }
```

Aboba & Lidyard

[Page 5]
std-encoding = (*:" " *SP value )
base-64-encoding = (*:" " *SP base64-value )
base64-value = <base-64-encoded value, defined in [11]>
sub-attr-encoding = *(";" sub-attr ":" <value>)
attr = radattr / snmpattr / tacacsplusattr / l2tpattr
radattr = *("RADIUS/"[^/]*) radattrname
radattrname = <a RADIUS attribute name, as defined in [3],[4]> / <a RADIUS attribute number, as defined in [3],[4]>
snmpattr = *("SNMP/"[^/]*) snmpattrname
snmpattrname = <an SNMP object ID> / <an SNMP attribute name>
tacacsplusattr = *("TACACS+//"[^/]*) tacacsplusattrname
tacacsplusattrname = <a TACACS+ attribute name, as defined in [13]> / <an L2TP attribute number, as defined in [13]>
l2tpattr = *("L2TP//"[^/]*) l2tpattrname
l2tpattrname = <an L2TP attribute number, as defined in [13]>
value = 1*safe-initval *safe
safe = <ASCII values 040 - 0176 octal (32 - 126 decimal), excluding semi-colon (";", ASCII 59 decimal)
safe-initval = <ASCII values 040 - 0176 octal (32 - 126 decimal), excluding colon (":", ASCII 58 decimal), SPACE, and semi-colon (";", ASCII 59 decimal)
SP = %x20 ; Space character
SEP = (CR LF) / LF
CR = <ASCII CR, carriage return>
LF = <ASCII LF, line feed>
Alpha = %x41-5A / %x61-7A ; A-Z / a-z
Digit = %x30-39 ; 0-9

5. Acknowledgements

Thanks to Glen Zorn of Microsoft and Pat Calhoun of Sun Microsystems for useful discussions of this problem space.

6. References


7. Authors’ Addresses

Bernard Aboba
Microsoft Corporation
One Microsoft Way
Redmond, WA 98052

Phone: 425-936-6605
EMail: bernarda@microsoft.com

Dave Lidyard
Telco Research Corporation
616 Marriott Drive
Nashville, TN 37214
Phone: 615-231-6110
EMail: dave@telcores.com