Avaya Call Management System

Call History Interface

Release 16.1
June 2010

Notice

While reasonable efforts were made to ensure that the information in this document was complete and accurate at the time of printing, Avaya Inc. can assume no liability for any errors. Changes and corrections to the information in this document might be incorporated in future releases.

Documentation disclaimer

Avaya Inc. is not responsible for any modifications, additions, or deletions to the original published version of this documentation unless such modifications, additions, or deletions were performed by Avaya. Customer and/or End User agree to indemnify and hold harmless Avaya, Avaya's agents, servants and employees against all claims, lawsuits, demands and judgments arising out of, or in connection with, subsequent modifications, additions or deletions to this documentation to the extent made by the Customer or End User.

Link disclaimer

Avaya Inc. is not responsible for the contents or reliability of any linked Web sites referenced elsewhere within this documentation, and Avaya does not necessarily endorse the products, services, or information described or offered within them. We cannot guarantee that these links will work all the time and we have no control over the availability of the linked pages.

Warranty

Avaya Inc. provides a limited warranty on this product. Refer to your sales agreement to establish the terms of the limited warranty. In addition, Avaya's standard warranty language, as well as information regarding support for this product, while under warranty, is available through the Avaya Support Web site:

http://www.avaya.com/support

License

USE OR INSTALLATION OF THE PRODUCT INDICATES THE END USER'S ACCEPTANCE OF THE TERMS SET FORTH HEREIN AND THE GENERAL LICENSE TERMS AVAILABLE ON THE AVAYA WEB SITE http://www.avaya.com/support/LicenseInfo ("GENERAL LICENSE TERMS"). IF YOU DO NOT WISH TO BE BOUND BY THESE TERMS, YOU MUST RETURN THE PRODUCT(S) TO THE POINT OF PURCHASE WITHIN TEN (10) DAYS OF DELIVERY FOR A REFUND OR CREDIT.

Avaya grants End User a license within the scope of the license types described below. The applicable number of licenses and units of capacity for which the license is granted will be one (1), unless a different number of licenses or units of capacity is specified in the Documentation or other materials available to End User. "Designated Processor" means a single stand-alone computing device. "Server" means a Designated Processor that hosts a software application to be accessed by multiple users. "Software" means the computer programs in object code, originally licensed by Avaya and ultimately utilized by End User, whether as stand-alone Products or pre-installed on Hardware. "Hardware" means the standard hardware Products, originally sold by Avaya and ultimately utilized by End User.

License type(s)

Designated System(s) License (DS). End User may install and use each copy of the Software on only one Designated Processor, unless a different number of Designated Processors is indicated in the Documentation or other materials available to End User. Avaya may require the Designated Processor(s) to be identified by type, serial number, feature key, location or other specific designation, or to be provided by End User to Avaya through electronic means established by Avaya specifically for this purpose.

Concurrent User License (CU). End User may install and use the Software on multiple Designated Processors or one or more Servers, so long as only the licensed number of Units are accessing and using the Software at any given time. A "Unit" means the unit on which Avaya, at its sole discretion, bases the pricing of its licenses and can be, without limitation, an agent, port or user, an e-mail or voice mail account in the name of a person or corporate function (e.g., webmaster or helpdesk), or a directory entry in the administrative database utilized by the Product that permits one user to interface with the Software. Units may be linked to a specific, identified Server.

Copyright

Except where expressly stated otherwise, the Product is protected by copyright and other laws respecting proprietary rights. Unauthorized reproduction, transfer, and or use can be a criminal, as well as a civil, offense under the applicable law.

Third-party components

Certain software programs or portions thereof included in the Product may contain software distributed under third party agreements ("Third Party Components"), which may contain terms that expand or limit rights to use certain portions of the Product ("Third Party Terms"). Information identifying Third Party Components and the Third Party Terms that apply to them is available on the Avaya Support Web site:

http://www.avaya.com/support/ThirdPartyLicense

Preventing toll fraud

"Toll fraud" is the unauthorized use of your telecommunications system by an unauthorized party (for example, a person who is not a corporate employee, agent, subcontractor, or is not working on your company's behalf). Be aware that there can be a risk of toll fraud associated with your system and that, if toll fraud occurs, it can result in substantial additional charges for your telecommunications services.

Avaya fraud intervention

If you suspect that you are being victimized by toll fraud and you need technical assistance or support, call Technical Service Center Toll Fraud Intervention Hotline at +1-800-643-2353 for the United States and Canada. For additional support telephone numbers, see the Avaya Support Web site:

http://www.avaya.com/support

Trademarks

Avaya and the Avaya logo are either registered trademarks or trademarks of Avaya Inc. in the United States of America and/or other jurisdictions.

All other trademarks are the property of their respective owners.

Downloading documents

For the most current versions of documentation, see the Avaya Support Web site:

http://www.avaya.com/support

Avaya support

Avaya provides a telephone number for you to use to report problems or to ask questions about your product. The support telephone number is 1-800-242-2121 in the United States. For additional support telephone numbers, see the Avaya Support Web site:

http://www.avaya.com/support

Contents

Preface

Avaya Call Management System (CMS) is an application for businesses and organizations that use Avaya communication servers to process large volumes of telephone calls using the Automatic Call Distribution (ACD) feature. Avaya CMS supports solutions for routing and agent selection, multi-site contact centers, remote agents, reporting, interfaces to other systems, workforce management, desktop applications, system recovery, and quality monitoring.

Avaya CMS is part of the Operational Effectiveness solution of the Avaya Customer Interaction Suite.

This section includes the following topics:

Purpose

The purpose of this document is to describe the format of the Call History data files, and how to transfer these files to another computer. Call History Interface (CHI) customers are responsible for the storage, formatting, printing, and any additional processing of the data transferred to the computer.

Intended users

This document is written for:

- Avaya support personnel
- Contact center administrators

Users of this document must be familiar with Avaya CMS.

Overview

This document includes the following topics:

- Overview of Internal Call History Interface on page 11

  Provides an overview of the Internal call history Interface.

- Overview of External Call History Interface on page 13

  Provides an overview of the External call history Interface.

- Call record formats on page 15

  Provides description about formats of the CMS call record files that are transferred by External

- Setting up the ECHI package on page 57

  Outlines the procedures for installing the External Call History Interface software and connecting the CMS computer to the receiving computer.

Conventions and terminology

If you see any of the following safety labels in this document, take careful note of the information presented.

CAUTION:
Caution statements call attention to situations that can result in harm to software, loss of data, or an interruption in service.

WARNING:
Warning statements call attention to situations that can result in harm to hardware or equipment.

DANGER:
Danger statements call attention to situations that can result in harm to personnel.

SECURITY ALERT:
Security alert statements call attention to situations that can increase the potential for unauthorized use of a telecommunications system.

Reasons for reissue

This document contains changes for updations to the call record formats and the example call records.

Documentation Web sites

All CMS documentation can be found at http://www.avaya.com/support. New issues of CMS documentation will be placed on this Web site when available.

Use the following Web sites to view related support documentation:

- Information about Avaya products and service

  http://www.avaya.com

- Sun hardware documentation

  http://docs.sun.com

Support

Contacting Avaya technical support

Avaya provides support telephone numbers for you to report problems or ask questions about your product.

For United States support:

1- 800- 242-2121

For international support:

See the Support Directory listings on the Avaya Web site.

Escalating a technical support issue

Avaya Global Services Escalation Management provides the means to escalate urgent service issues.

Overview of Internal Call History Interface

The Internal Call History Interface (ICHI) feature is the default call history recording application that is delivered with CMS. If you do not want to use the External Call History Interface (ECHI) feature package, you can use ICHI to collect and process call records directly from the Informix database using applications such as ODBC. As with ECHI, you must allocate space to store the records.

Note:
You cannot use both ICHI recording and ECHI recording at the same time.

This section includes the following topics:

Space allocation

You need to allocate space with Data Storage Allocation (DSA). You must use DSA to administer the number of call records stored for each ACD. The sum total of call records which can be stored is shown in the CMS limits documented in *Avaya Aura™ Communication Manager System Capacities Table*. After the call record limit is reached, CMS automatically begins deleting the oldest records in the table to make room for newer records. The CMS Maintenance Backup will back up the **call_rec** table, which can be restored or migrated. You may also create custom reports from the **call_rec** table using CMS custom reports or CMS Supervisor Report Designer.

Note:
In addition to using custom and designer reports, you can also use ODBC/JDBC to pull the **call_rec** records to an external client for reporting.

The rate of writing Internal Call History (ICH) records to the database table is limited to 4,000 within a 20-minute interval.

Call record processing

Call records are stored in the call_rec table if the following conditions are true:

- The master ACD link must be operational.
- The value for the **Number of call records** field in Data Storage Allocation must be greater than zero.

About the call record format

The call record field descriptions are the same for ICH and ECH. For more information please refer to Call record field descriptions on page 16.

Overview of External Call History Interface

The External Call History Interface (ECHI) feature allows you to transfer the CMS call record data to another computer for processing. This feature is especially useful for call centers with a high volume of calls.

Note:
You cannot use both ICHI recording and ECHI recording at the same time.

This section includes the following topics:

Space allocation

You need to allocate space with Data Storage Allocation (DSA). You must use DSA to administer the number of call records stored for each ACD. The sum total of call records which can be stored is shown in the CMS limits documented in *Avaya Aura™ Communication Manager System Capacities Table*. After the call record limit is reached, CMS begins deleting the oldest files not extracted from the `/cms/cmstables` directory.

Call record processing

As calls are processed, call records are recorded in a file (one record per call segment) until either the file size reaches approximately 1 MB or at the end of an interval. Then the file of call record segments will be transferred to another computer via uucp or other designated software.

The following requirements must be met for ECHI to transfer the files at the end of an interval:

- The master ACD link must be operational.
- For the master ACD, the value for the **Number of call records** field in Data Storage Allocation must be greater than zero.

If the above conditions are not met, the files will not be transferred until they reach the maximum size.

If the computers are in close proximity, the data is transferred at speeds of up to 38,400 bps. A new file and its first record will be created as the first segment of the next call is processed.

About the call record file format

The Call record field descriptions are transferred by the Call History Interface feature. For more information please refer to Call record formats on page 15.

Call record formats

This section describes the format of the CMS call record files that are transferred by External Call History Interface (ECHI).

This section includes the following information:

Data characteristics

Integers are stored with the least significant byte first and the most significant byte last. All integers are unsigned.

Data items of type char are stored with the most significant byte first and the least significant byte last. Type char data items are ASCII character strings of the length specified in the Length column.

Note:

The following sections explain how CMS sends integers to the receiving computer. The receiving computer may have different descriptions for integers. Once your software has parsed and stored the data on the receiving computer, it may be transformed to a different representation.

File name format

The call history files are named *chrxxxx* (*chr* followed by a 4-digit number) on the CMS computer. The transfer process adds an extension of up to three digits so that the file name on the receiving computer is *chrxxxx.x, chrxxxx.xx or chrxxxx.xxx*.

File header

Each call record file header includes a **VERSION** field and a **SEQUENCE _NUMBER** field.

When using the NICE Analyzer with CMS, you need to inform the NICE Analyzer of new call record formats through the version field in the External Call History file header and the External Call History Call Record header.

This section includes the following topics:

About the VERSION field

The **VERSION** field is a 4-byte Integer.

The **VERSION** field will contain the value 16 for every file header.

About the SEQUENCE _NUMBER

The **SEQUENCE _NUMBER** field is a 4-byte Integer. The **SEQUENCE _NUMBER** field identifies a particular call record file so that duplicates can be recognized when retransmission has occurred. These files are sequential. The **SEQUENCE_NUMBER** restarts at zero when its 4-byte integer reaches its limit (when all bits are high [ones] in its binary equivalent).

Example call record file header

The following table is an example of the information contained in a call record header. The least significant bytes are stored first and most significant bytes are stored last.

	VERSION	SEQUENCE_NUMBER
Decimal	16	1
Binary	00010000 00000000 00000000 00000000	00000001 00000000 00000000 00000000

Call record field descriptions

The file header is followed by a file containing a variable number of fixed-length records. Call History Interface does not use CMS database items. The call record data items are represented by their CMS database equivalents. If a field is not measured, the call record field will be populated with a "null", “0”, or "-1".

ACD

Type: integer

Length: 1 byte

Description: The ACD number for which data was collected.

ACWTIME

Type: integer

Length: 4 bytes

Description: The time spent, in seconds, in After Call Work (ACW) associated with this call by the answering agent in this segment.

AGENTSKILLLEVEL

Type: integer

Length: 1 byte

Description: Level assigned to the agent for the skill the call is delivered to.

AGENTSURPLUS

Type: integer

Length: 1 byte

Description: Whether the call is delivered under agent surplus or call surplus condition.

- 0 = NA
- 1 = Call surplus: ACD call was routed to the agent after waiting in the queue.
- 2 = Agent surplus: ACD call was routed to the agent without waiting in the queue.

AGT_RELEASED

Type: bit

Length: 1 bit

Description: The agent released or dropped the split/skill or direct agent ACD call. This is always true for ACD calls the agent transferred or conferenced. (0=NO, 1=YES). This value is padded with seven 0s to maintain byte alignment.

ANS_LOCID

Type: integer

Length: 2 bytes

Description: The location ID of the answering agent. This ID number is not assigned to an agent, but rather to the agent terminal and is associated with the communication server port network ID. An agent may be associated with a location ID only upon logging into the ACD. Valid values are 0 through 250.

Important:
If the location IDs (LOC_ID) defined on the Communication Manager server are greater than the valid CMS values, then a default location ID of 0 will be assigned.

ANSHOLDTIME

Type: integer

Length: 4 bytes

Description: The total time, in seconds, the call was put on hold by the answering agent in this call segment. In agent-to-agent calls, **ANSHOLDTIME** is accrued for the answering agent if the agent puts the call on hold, but not for the other agent (who continues to accrue talk time).

Hold time is accrued for any type of call.

ANSLOGIN

Type: char

Length: 16 bytes

Description: The login ID of the agent who answered the call in this segment. This field is blank for unmeasured extensions when EAS is not active.

ANSREASON

Type: integer

Length: 1 byte

Description: The reason code associated with the answering agent's mode, if the agent is in the AUX mode. For agents in AUX for communication servers that do not have EAS and reason codes active, **ANSREASON** is always 0.

ASAIUUI

Type: unsigned char

Length: 97 bytes

Description: The last ASAI user-to-user information associated with a call segment.

If an **ASAIUUI** is not sent, then this field will be NULL.

ASSIST

Type: bit

Length: 1 bit

Description: Whether the answering agent in this segment requested supervisor assistance on this call. Valid values are 0=NO, 1=YES.

AUDIO

Type: bit

Length: 1 bit

Description: Whether an agent in this segment reported an audio difficulty problem. Valid values are 0=NO, 1=YES.

CALL_DISP

Type: integer

Length: 1 byte

Description: This field represents the call disposition and indicates whether the call in the segment was:

- **1=connected** (CONN, non-ACD call to a measured agent)

  A connected call is a non-ACD call to a measured agent for which CMS receives an indication that the call was connected.

- **2=answered** (ANS, split/skill or direct agent call answered by an agent)

  An answered call is any split/skill or direct agent ACD call for which CMS receives an indication that the call was answered by an agent and was not a phantom abandon.

- **3=abandoned** (ABAN)

  An abandoned call is any ACD call in which a caller hangs up before receiving an answer from an agent and for which CMS receives notification that the caller abandoned. Phantom abandons (**PHANTOMABNS**) are included as abandoned calls.

- **4=interflowed** (IFLOW)

  Interflowed calls are calls that are interflowed to an off-switch destination.

- **5=forced busy** (FBUSY)

  Forced busy calls are calls that CMS records as **BUSYCALLS** for the trunk group that carried them.

  These calls can be VDN calls that received a forced busy from the vector command.

- **6=forced disconnect** (FDISC)

  Forced disconnect calls are VDN calls that are disconnected by the communication server due to the execution of a disconnect vector command.

- **7=other** (OTHER)

  Forced disconnect calls also include calls disconnected because of the vector disconnect timer or because they reached the end of vector processing without being queued.

  Other calls include any other calls that do not fall into categories such as answered or abandoned. See definitions for individual tables for **OTHERCALLS**.

CALLID

Type: integer

Length: 4 bytes

Description: A unique number assigned to this call and all its call segments. For conferenced/transferred calls, two (or more) calls are tied together. When the entire call is recorded, one call ID is used to tie together all call segments. In “meet-me” conferences, this may result in a “later” segment of the call starting earlier than the first segment. Call IDs are not necessarily strictly sequential, but will be unique for calls over a day. For additional information on “meet-me” conferences, see the appropriate Avaya Communication Manager administrator guide.

CALLING_II

Type: char

Length: 3 bytes

Description: The Information Indicator (II) digits associated with the call. These digits are a two-digit string provided by ISDN Primary Rate Interface (PRI) to indicate the type of originating line of the caller. These digits supply information about the originator location, for example, pay phone, hospital, or prison. The column is blank if the call does not contain II digits.

CALLING_PTY

Type: char

Length: 25 bytes

Description: The calling party identification, (which is the Automatic Number Identification (ANI)/Station Identification (SID) for Integrated Services Digital Network (ISDN) ANI delivery), extension or trunk equipment location identifying the originator of the call. This field is blank if the trunk is not measured, or for internal calls if the originating extension is not measured. (Up to 24 digits in this field.)

CONFERENCE

Type: bit

Length: 1 bit

Description: Whether the answering agent initiated a conference on this segment. Valid values are 0=NO, 1=YES.

CONSULTTIME

Type: integer

Length: 4 bytes

Description: The time an agent talked on any outbound call while in AUX work, ACW, or in OTHER with a call on hold. This includes the time the originating agent spent talking to the destination party while establishing a conference or transferring a call. (This is the time between presses of the transfer or conference button.) It includes wait time if the agent is calling a Vector Directory Number (VDN) or split/skill extension, but the wait time can be obtained by subtracting the **DISPTIME** item from **CONSULTTIME**.

CWC1 through CWC5

Type: char

Length: 17 bytes

Description: The Call Work Code entered by an agent for the Call Segment.

The first five Call Work Codes entered by an agent are stored in the call segment of a call record. The last Call Work Code entered by an agent for a call segment will continue to be stored in the LASTCWC column. The following data items contain Call Work Codes 1 through 5:

- **CWC1**
- **CWC2**

- **CWC3**
- **CWC4**
- **CWC5**

DA_QUEUED

Type: bit

Length: 1 bit

Description: Whether the call was queued as a direct agent call Valid values are 0=NO, 1=YES.

DIALED_NUM

Type: char

Length: 25 bytes

Description: The number the caller dialed (up to 24 digits). This will be the VDN for inbound vectoring calls, blank for inbound calls without vectoring, and dialed digits for outbound calls.

DISPIVECTOR

Type: integer

Length: 16 bytes

Description: The number of the first vector associated with the disposition VDN (DISPVDN).

DISPPRIORITY

Type: integer

Length: 1 byte

Description: The priority the call had at its disposition in this segment. Priorities can be 1=NO or 2=YES (without vectoring), or 3=LOW, 4=MED, 5=HIGH, or 6=TOP (with vectoring). If the call never gets queued to a split/skill, the priority will not be set.

DISPSKLEVEL

Type: integer

Length: 1 byte

Description: The skill level (1 through 16) associated with the skill for which the agent answered the call or for calls that abandoned from ringing or from a direct agent queue with the agent from whom the call abandoned.

DISPSPLIT

Type: integer

Length: 2 bytes

Description: The number of the split/skill associated with the call at its disposition in this call segment. Calls that were not queued to a split or skill at the time of disposition will have **DISPSPLIT** set to null. Calls that were queued to an unmeasured split/skill at the time of disposition will have **DISPSPLIT** set to zero.

DISPTIME

Type: integer

Length: 4 bytes

Description: The wait time (in the vector, in queue, and ringing) until the disposition is recorded in **CALL_DISP** for the segment. For extension calls made directly to agents (not through a VDN), this will always be zero.

DISPVDN

Type: char

Length: 16 bytes

Description: The number of the VDN associated with the call at its disposition for this call segment. **DISPVDN** will be blank for calls that are not associated with a VDN at their disposition.

DURATION

Type: integer

Length: 4 bytes

Description: The total time the trunk was in use. An alternate description would be the length of time the trunk has been in the TK state. This is the overall trunk holding time from the beginning of the call segment until the caller is disconnected. For the first segment of a call, this will be the trunk holding time for the caller for the entire call (from seized until idle). With a transfer, the original trunk remains associated with both call segments until the call ends.

EQ_LOCID

Type: integer

Length: 2 bytes

Description: The location ID of the trunk. This ID number is not assigned directly to a trunk, but rather to the communication server port network. Therefore, each trunk on the network will have the same location ID number. Valid values are ID numbers 0 through 250.

EQLOC

Type: char

Length: 10 bytes

Description: The physical equipment location (trunk number) for which data was collected or for which the exception occurred. This will be blank if the trunk is not measured. The value for this field is eight characters followed by two null characters.

EVENT1-9

Type: integer

Length: 1 byte each

Description: The number of times each event (stroke count) button (buttons 1 to 9) was entered for this call segment.

FIRSTVDN

Type: char

Length: 16 bytes

Description: The number of the first VDN associated with the call segment. This will be blank for calls not associated with a VDN.

FIRSTVECTOR

Type: integer

Length: 2 bytes

Description: The number of the first vector associated with the first VDN for the call segment. This will be blank if no vector is involved.

HELD

Type: integer

Length: 1 byte

Description: The total number of times this call was placed on hold by the answering agent in this call segment. With agent-to-agent calls, this count is incremented for the agent who puts the call on hold, but not for the calling agent.

HOLDABN

Type: bit

Length: 1 bit

Description: Whether this on-hold call was abandoned from hold in this call segment. Valid values are 0=NO, 1=YES.

INTERRUPTDEL

Type: Interger

Length: 1 byte

Description: Whether the call is delivered when agent is interrupted from an interruptible aux state.

LASTCWC

Type: char

Length: 17 bytes

Description: The last call work code (up to 16 digits) entered by the answering agent in this segment.

LASTDIGITS

Type: char

Length: 17 bytes

Description: The last set of collected digits sent to the CMS by the communication server for this call. These are digits the communication server sends to CMS when it executes a “collect” vector command. The digits may be digits the caller was prompted to enter, either through the prompting feature on the communication server or through network-prompted digits [caller-entered digits CED], customer database-provided digits (CDPD from the network), or digits collected through a “converse” vector command.

LASTOBSERVER

Type: char

Length: 16 bytes

Description: The login ID of the last agent who service-observed or bridged on to this call.

MALICIOUS

Type: bit

Length: 1 bit

Description: Whether a malicious call trace was activated for this call segment. Valid values are 0=NO, 1=YES.

NETINTIME

Type: integer

Length: 4 bytes

Description: The time the call spent in a VDN processing at another communication server located elsewhere in the network.

OBS_LOCID

Type: integer

Length: 2 bytes

Description: The location ID of the observing agent. This ID number is not assigned to an agent, but rather to the agent terminal and is associated with the communication server port network ID. An agent may be associated with a location ID only upon login to the ACD. Valid values are ID numbers from 0 through 250.

OBSERVINGCALL

Type: bit

Length: 1 bit

Description: Whether this call represents an agent observing or bridging on to an existing call. Valid values for **OBSERVINGCALL** are 0=NO, 1=YES. Some reports will display only a 1 (YES).

ORIG_LOCID

Type: integer

Length: 2 bytes

Description: The location ID of the calling agent. This ID number is not assigned to an agent, but rather to the agent terminal and it is associated with the communication server port network ID. An agent may be associated with a location ID only upon logging into the ACD. Valid values are ID numbers from 0 through 250.

ORIGHOLDTIME

Type: integer

Length: 4 bytes

Description: The total time the call was put on hold by the originating agent.

ORIGLOGIN

Type: char

Length: 16 bytes

Description: The login ID of the agent originating the call. This is used for calls an agent originates to another agent, to an on-switch extension, or to an external destination.

ORIGREASON

Type: integer

Length: 1 byte

Description: The reason code associated with the originating agent’s mode, if the agent is in the AUX mode.

PREFSKILLLEVEL

Type: integer

Length: 1 byte

Description: Whether the call is delivered via the preferred skill level check vector command.

- 0 = NA: Call was not delivered via the preferred skill level command or no skill preference was specified.
- 1 = not preferred: Agent's level for the skill does not match the level specified in the check vector command.
- 2 = preferred: Agent's level for the skill matches the preferred skill level specified by the check vector command.

QUEUETIME

Type: integer

Length: 4 bytes

Description: The time a call spends in queue for a call segment

QUEUETIME is the time from when a call first queues to a skill until it starts ringing at an agent. If a call consists of multiple segments, the queue time includes the time associated with that segment. The default is 0 if there is no queue time. **QUEUETIME** is stored for all Avaya CMS supported communication server releases.

RINGTIME

Type: integer

Length: 4 bytes

Description: The time a call spends ringing at an agent station for a call segment **RINGTIME** includes all ring time. This is the time a call spends ringing at an agent position and is independent of the final disposition of the call and ring time associated with RONA. If a call consists of multiple segments, each segment contains its associated ring time. The default is 0 if there is no ring time. **RINGTIME** is stored for all Avaya CMS supported communication server releases.

SEGMENT

Type: integer

Length: 1 byte

Description: The number identifying the call segment. Segment numbers are from 1 up to the number of segments in the call.

SEGSTART

Type: integer

Length: 4 bytes

Description: The UNIX time and date when the call segment started. Call segments start when CMS receives the first message for the call, since each call segment represents a call. (When an agent transfers or conferences a call, the agent makes another call to bring about the transfer/conference.)

SEGSTART_UTC

Type: integer

Length: 4 bytes

Description: SEGSTART time (UNIX time and date) adjusted to be Coordinated Universal Time. See SEGSTART for details.

SEGSTOP

Type: integer

Length: 4 bytes

Description: The UNIX time and date when the call segment ended. A call segment ends when all trunks and agents associated with the call segment have dropped off the call. This means that after call work time for the agents is included when calculating the call segment stop time.

SEGSTOP_UTC

Type: integer

Length: 2 bytes

Description: SEGSTOP time (UNIX time and date) adjusted to be Coordinated Universal Time. See SEGSTOP for details.

SPLIT1

Type: integer

Length: 2 bytes

Description: The first split/skill the call queued to in the first VDN with which it was associated in the call segment.

SPLIT2

Type: integer

Length: 2 bytes

Description: The second split/skill the call was also queued to in the first VDN with which it was associated in the call segment.

SPLIT3

Type: integer

Length: 2 bytes

Description: The third split/skill the call was also queued to in the first VDN with which it was associated in the call segment.

TALKTIME

Type: integer

Length: 4 bytes

Description: The total talk time for the answering agent in this segment.

TKGRP

Type: integer

Length: 2 bytes

Description: The trunk group number for which data was collected (or for which an exception occurred). This will be null if the trunk group carrying the call is not measured.

TRANSFERRED

Type: bit

Length: 1 bit

Description: Whether an answering agent initiated a transfer for this call segment. Valid values are 0=NO, 1=YES.

UCID

Type: char

Length: 21 bytes

Description: The Universal Call Identifier - a unique number assigned to this call segment within the customer network.

UUI_LEN

Type: short integer

Length: 2 bytes

Description: The length of the UUI information in bytes

If an **ASAIUUI** is not provided by the communication server, the default length is 0.

VDN2 through VDN9

Type: char

Length: 16 bytes

Description: The first 9 VDNs and the last VDN associated with a call segment.

The existing **FIRSTVDN** data item contains the first VDN and **DISPVDN** contains the final VDN. The following data items contain VDNs 2 through 9:

- **VDN2**
- **VDN3**
- **VDN4**
- **VDN5**
- **VDN6**
- **VDN7**
- **VDN8**
- **VDN9**

VDN2 through **VDN9** are populated only when a call touches more than one measured VDN. Data items that are not populated are NULL. **DISPVDN** is usually populated with the same values as the last VDN populated. If the last VDN populated was **VDN7**, then **DISPVDN** will include the same values as **VDN7**. The exception to this rule is when there are 10 or more VDNs associated with a call.

Example call scenario and call records

This section presents an example of a record scenario in record format. In this example an inbound vectored call is answered by an ACD agent and is then transferred to a VDN. After the transfer, two call records are generated. One call record is generated for every call segment.

Note:
The tables in the following examples do not include any of the fields which were used in the extended ECH or the fields added with R16.

Important:
These record scenarios are examples of what a typical call record would look like. The call records you see for your location might vary from this example.

This section includes:

First call segment

The following occurs during the first call segment:

1. A call comes into ACD 1 on TG 32 (carried on the trunk located at 0101A0102) to VDN 43211 which points to Vector 33 at 07:37:10 on 04/16/02.
2. The call queues to Split 1 at medium priority via a “queue to” command, and then queues to backup Split 2 at low priority via a “backup” command.
3. The call waits 10 seconds, then rings for 5 seconds, and then is answered by agent 5018 in Split 1.
4. The caller and agent talk for 44 seconds. The agent transfers the call to VDN 43712 which points to Vector 37. (The call is held for 4 seconds while transferring.)
5. The agent has 42 seconds of after call work during which the agent enters call work code 12345.

First call segment record

The first column in the First call segment record table on page 33 shows whether the field is part of the header or a record field. See Call record field descriptions on page 16 for the field type, field length and field description. The Call report entry column shows data that would appear on a CMS call report for this call segment. The Call record entry column shows the binary equivalent of the Call report entry data. This binary data appears in an actual call record file in the order shown in the following table. Spaces are shown between bytes for clarity in the Call record entry column.

First call segment record table

Field	Call report entry	Call record entry
VERSION (header)	16	00010000 00000000 00000000 00000000
SEQUENCE_NUM (header)	1	00000001 00000000 00000000 00000000
CALLID	212	11010100 00000000 00000000 00000000
ACWTIME	00:42	00101010 00000000 00000000 00000000
ANSHOLDTIME	00:04	00000100 00000000 00000000 00000000
CONSULTTIME	00:00	00000000 00000000 00000000 00000000
DISPTIME	00:15	00001111 00000000 00000000 00000000

First call segment record table (continued)

Field	Call report entry	Call record entry
DURATION	04:25	00001001 00000001 00000000 00000000
SEGSTART	07:37:10 04/16/02	10100110 11010100 10111011 00111100
SEGSTART_UTC	14:37:10 04/16/02	00010110 00110111 10111010 00111100
SEGSTOP	07:42:35 04/16/02	11101011 11010101 10111100 00111100
SEGSTOP_UTC	14:42:35 04/16/02	01011011 00111000 10111010 00111100
TALKTIME	00:44	00101100 00000000 00000000 00000000
NETINTIME	98	01100010 00000000 00000000 00000000
ORIGHOLDTIME	55	00110111 00000000 00000000 00000000
QUEUETIME	10	00001010 00000000 00000000 00000000
RINGTIME	5	00000101 00000000 00000000 00000000

First call segment record table (continued)

Field	Call report entry	Call record entry
DISPIVECTOR	37	00100101 00000000
DISPSPLIT	1	00000001 00000000
FIRSTIVECTOR	37	00100101 00000000
SPLIT1	1	00000001 00000000
SPLIT2	2	00000010 00000000
SPLIT3	0	00000000 00000000
TKGRP	32	00100000 00000000
EQ_LOCID	33	00100001 00000000
ORIG_LOCID	13	00001101 00000000
ANS_LOCID	23	00010111 00000000
OBS_LOCID	42	00101010 00000000
UUI_LEN	0	00000000 00000000
ASSIST	N	0
AUDIO	N	0
CONFERENCE	N	0
DA_QUEUED	N	0
HOLDABN	N	0
MALICIOUS	N	0
OBSERVINGCALL	Y	1
TRANSFERRED	Y	1

First call segment record table (continued)

Field	Call report entry	Call record entry
AGT_RELEASED	Y	10000000
ACD	1	00000001
CALL_DISP	Ans	00000010
DISPPRIORITY	4 (MED)	00000100
HELD	1	00000001
SEGMENT	1	00000001
ANSREASON	0	00000000
ORIGREASON	0	00000000
DISPSKLEVEL	3	00000011
EVENT1-9	0	00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000
UCID	0006300089088 5813486	00000000 00000000 00000000 00000110 00000011 00000000 00000000 00000000 00001000 00001001 00000000 00001000 00001000 00000101 00001000 00000001 00000011 00000100 00001000 00000110 00000000

First call segment record table (continued)

Field	Call report entry	Call record entry
DISPVDN	43211	00110100 00110011 00110010 00110001 00110001 00110000 00110000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000
EQLOC	0101A01	00110000 00110001 00110000 00110001 01000001 00110000 00110001 00110000 00000000 00000000
FIRSTVDN	43211	00110100 00110011 00110010 00110001 00110001 00110000 00110000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000

First call segment record table (continued)

Field	Call report entry	Call record entry
ORIGLOGIN	5011	00110101 00110000 00110001 00110001 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000
ANSLOGIN	5018	00110101 00110000 00110001 00111000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000

First call segment record table (continued)

Field	Call report entry	Call record entry
LASTOBSERVER	5013	00110101 00110000 00110001 00110011 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000
DIALED_NUM	43211	00110100 00110011 00110010 00110001 00110001 00000000

First call segment record table (continued)

Field	Call report entry	Call record entry
CALLING_PTY	010170102	00110000 00110001 00110000 00110001 00110111 00110000 00110001 00110000 00110010 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000
LASTDIGITS	1	00000001 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000

First call segment record table (continued)

Field	Call report entry	Call record entry
LASTCWC	12345	00110001 00110010 00110011 00110100 00110101 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000
CALLING_II	0	00000000 00000000 00000000
CWC1	12345	00110101 00110010 00110011 00110100 00110001 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000

First call segment record table (continued)

Field	Call report entry	Call record entry
CWC2	0	00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000
CWC3	0	00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000

First call segment record table (continued)

Field	Call report entry	Call record entry
CWC4	0	00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000
CWC5	0	00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000

First call segment record table (continued)

Field	Call report entry	Call record entry
VDN2-9	0	00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000
ASAI_UUI	0	193 bytes of 00000000
INTERRUPTDEL	0	00000000
AGENTSURPLUS	0	00000000
AGENTSKILLLEVEL	3	00000011
PREFSKILLLEVEL	0	00000000

Second call segment

The scenario continues with the second call segment of the inbound vectored call being transferred to a VDN. The call was transferred by agent 5018 to a VDN. The Second call segment record on page 45 explains the call record for the second call segment. This record has no header information because it follows the first call segment record in the same file.

The following occurs during the second call segment:

1. The caller is transferred to VDN 43712 which points to Vector 37.
2. The call is queued to Splits 4 and 5 at high priority using a “queue to” step to queue to Split 4, and a “check” step to queue to Split 5.
3. The call waits 1 second and then rings for 2 seconds before being answered by agent 2139 in Split 5.

4. The caller and agent talk for 3 minutes, 19 seconds, and then the agent releases the call.
5. The agent has 1 minute of after call work during which the agent enters call work code 67890. Stop time is 7:42:35 (includes the ACW time that extends beyond the time at which the caller dropped).

Second call segment record

The first column in the Second call segment record table on page 45 shows that all fields in this record are record fields. See Call record field descriptions on page 16 of this document for the field type, length, and description. The Call report entry column shows data that would appear on a CMS call report for this call segment. The Call record entry column shows the binary equivalent of the Call report entry data. This binary data appears in an actual call record file in the order shown in the following table. Spaces are shown between bytes for clarity in the Call record entry column.

Second call segment record table

Field	Call report entry	Call record entry
CALLID	212	11010100 00000000 00000000 00000000
ACWTIME	01:00	00111100 00000000 00000000 00000000
ANSHOLDTIME	00:00	00000000 00000000 00000000 00000000
CONSULTTIME	00:00	00000000 00000000 00000000 00000000
DISPTIME	00:03	00000011 00000000 00000000 00000000
DURATION	03:22	11001010 00000000 00000000 00000000

Second call segment record table (continued)

Field	Call report entry	Call record entry
SEGSTART	07:38:13 04/16/02	11100101 11010100 10111011 00111100
SEGSTART_UTC	14:38:13 04/16/02	01010101 00110111 10111100 00111100
SEGSTOP	07:42:35 04/16/02	01010101 11010101 10111011 00111100
SEGSTOP_UTC	14:42:35 04/16/02	01011011 00111000 10111100 00111100
TALKTIME	03:19	11000111 00000000 00000000 00000000
NETINTIME	98	01100010 00000000 00000000 00000000
ORIGHOLDTIME	55	00110111 00000000 00000000 00000000
QUEUETIME	1	0000010 00000000 00000000 00000000
RINGTIME	2	00000010 00000000 00000000 00000000
DISPIVECTOR	37	00100101 00000000
DISPSPLIT	5	00000101 00000000

Second call segment record table (continued)

Field	Call report entry	Call record entry
FIRSTIVECTOR	37	00100101 00000000
SPLIT1	4	00000100 00000000
SPLIT2	5	00000101 00000000
SPLIT3	0	00000000 00000000
TKGRP	32	00100000 00000000
EQ_LOCID	33	00100001 00000000
ORIG_LOCID	23	00010111 00000000
ANS_LOCID	23	00010111 00000000
OBS_LOCID	0	00000000 00000000
UUI_LEN	0	00000000 00000000
ASSIST	N	0
AUDIO	N	0
CONFERENCE	N	0
DA_QUEUED	N	0
HOLDABN	N	0
MALICIOUS	N	0
OBSERVINGCALL	Y	1
TRANSFERRED	Y	1
AGT_RELEASED	Y	10000000
ACD	1	00000001
CALL_DISP	Ans	00000010

Second call segment record table (continued)

Field	Call report entry	Call record entry
DISPPRIORITY	5 (HIGH)	00000101
HELD	0	00000000
SEGMENT	2	00000010
ANSREASON	1	00000001
ORIGREASON	9	00001001
DISPSKLEVEL	3	00000011
EVENT1-9	0	00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000
UCID	00063000890 885813486	00000000 00000000 00000000 00000110 00000011 00000000 00000000 00000000 00001000 00001001 00000000 00001000 00001000 00000101 00001000 00000001 00000011 00000100 00001000 00000110 00000000

Second call segment record table (continued)

Field	Call report entry	Call record entry
DISPVDN	43712	00110100 00110011 00110111 00110001 00110010 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000
EQLOC	0101A01	00110000 00110001 00110000 00110001 01000001 00110000 00110001 00110000 00000000 00000000
FIRSTVDN	43712	00110100 00110011 00110111 00110001 00110010 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000

Second call segment record table (continued)

Field	Call report entry	Call record entry
ORIGLOGIN	5018	00110101 00110000 00110001 00111000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000
ANSLOGIN	2139	00110010 00110001 00110011 00111001 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000

Second call segment record table (continued)

Field	Call report entry	Call record entry
LASTOBSERVER	0	00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000
DIALED_NUM	43712	00110100 00110011 00110111 00110001 00110010 00000000

Second call segment record table (continued)

Field	Call report entry	Call record entry
CALLING_PTY	0101A0102	00110000 00110001 00110000 00110001 01000001 00110000 00110001 00110000 00110010 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000
LASTDIGITS	0	00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000

Second call segment record table (continued)

Field	Call report entry	Call record entry
LASTCWC	67890	00110110 00110111 00111000 00111001 00110000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000
CALLING_II	0	00000000 00000000 00000000
CWC1	67890	00110110 00110111 00111000 00111001 00110000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000

Second call segment record table (continued)

Field	Call report entry	Call record entry
CWC2	0	00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000
CWC3	0	00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000

Second call segment record table (continued)

Field	Call report entry	Call record entry
CWC4	0	00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000
CWC5	0	00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000

Second call segment record table (continued)

Field	Call report entry	Call record entry
VDN2-9	0	00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000
ASAI_UUI	0	193 bytes of 00000000
INTERRUPTDEL	0	00000000
AGENTSURPLUS	0	00000000
AGENTSKILLLEVEL	3	00000011
PREFSKILLLEVEL	0	00000000

Setting up the ECHI package

This section describes how to install the External Call History Interface (ECHI) software, connect the CMS computer to the receiving computer, and select and set up the receiving computer.

Note:
If you are upgrading your CMS to a newer version, applications associated with ECHI might have to be rewritten because of changes to call record database items in CMS. Make sure the version field is correct. For more information, see Call record formats on page 15.

This section includes the following sections:

Customer responsibilities

You must perform the following steps to be able to transfer call history data:

- Purchase the ECHI package.
- Contact the Technical Service Center (TSC) to authorize the feature. If you are an international customer, contact your Avaya representative.
- Install the feature on CMS.
- Set up the connection between the CMS computer and the computer that will be receiving the data.
- Provide the receiving computer applications that will:
  - Allow the receiving computer to receive data via uucp or another file-transfer utility of your choosing.
  - Parse the files of data being transferred.
  - Store the data in some usable fashion (For example, in a database on the receiving computer).
  - Convert the data to a usable format.

Prerequisites

Before installation, be sure you have obtained authorization for the ECHI feature package.

Ports

At installation, the ECHI feature will automatically select the port it will use.

Installing the ECHI feature

CMS can only support Internal Call History or External Call History at one time. The two packages can not be used simultaneously. Once the External Call History Interface feature is installed, call data will no longer be inserted into the call_rec table and you will not be able to access the CMS Call Record report. NICE Analyzer is an optional package that allows you to view external call history data. Contact your Avaya representative for more information.

To install and set up ECHI on a Sun system:

1. Log in as **root**.

   The computer must be in run-level 2, and all file systems must be mounted.

2. Enter:

   cmssvc

   The system displays the `Avaya Call Management System Services Menu`.

3. Enter the number associated with the `auth_display` option and verify that the system is authorized to install the External Call History package.

   Note:
   If External Call History is not authorized but needs to be, call the Avaya National Customer Care Center at 1-800-242-2121. International customers should contact their local Avaya distributor or representative.

4. CMS must be turned off before the package can be installed. Enter:

   cmsadm

   The system displays the `Avaya Call Management System Administration Menu`.

5. Enter the number associated with the `run_cms` option.

6. Enter the number associated with the `Turn off CMS but Leave IDS running` option.

   The system displays the following message:

```
*** CMS is now off ***
```

7. Enter:

   `cmsadm`

   The system displays the `Avaya Call Management System Administration Menu`.

8. Enter the number associated with the `pkg_install` option.

   The system displays a list of the installed CMS features.

9. Enter the number that corresponds to `External Call History`.

   The system displays the following message:

```
Enter the name of the computer to which to send call records (up
to 256 characters):
```

10. Enter the name of the receiving computer.

    The system displays the following message:

```
Enter the full path of the program to transfer call history files
(default: /cms/dc/chr/uucp_copy):
```

Note:

Currently, CMS uses uucp to transfer External Call History files to a designated remote machine and uustat to check that the files were transferred successfully.

You may optionally change these default programs to be customer-specified file transfer and check programs such as:

- Transmission Control Protocol/Internet Protocol (TCP/IP)
- UNIX commands remote copy (rcp) or copy (cp) using a remote file system (RFS)

To use either communications program enter a different full file path instead of the default uucp/uustat interface path.

11. Enter the full path of the default file transfer utility.

    For example, **`/cms/dc/chr/uucp_copy`** or optionally enter the full path of your specific file-transfer program.

    The system displays the following message:

```
Enter the full path of the program to check the external call
history file transmission (default: /cms/dc/chr/uucp_check):
```

12. Enter the full path of the default file transmission check program.

    For example, **/cms/dc/chr/uucp_check** or optionally enter the full path of your specific transmission-checking program.

    Note:
    If you have chosen to enter your own file transfer and transmission checking programs, continue with Step 16, entering the number of call records in the buffer.

    If you have chosen the CMS defaults, continue with Step 13.

    The system displays the following message:

```
Enter password for nuucp login on XXXXXXXX (up to 8
  characters):
```

13. Enter the nuucp password for the receiving computer.

    The system displays the following message:

```
Enter CMS port for connection to XXXXXXXX (s-pdeXXXX):
```

14. Enter the port on the CMS computer to be used by the receiving computer. (For a Sun system, s_pdev*xxxx* will be displayed instead.)

    The system displays the following message:

```
Select a speed for this connection
1) 19200
2) 38400
```

15. Enter the number associated with the speed for the connection between the CMS and the receiving computer.

    The system displays the following message:

```
Number of call segments to buffer for ACD XXXXXXX (0-xxxxxx):
```

16. Enter the number of call records to be held in the buffer if the receiving machine cannot accept the data (minimum: 20 MB). This reserves disk space; therefore, sufficient disk space must be available.

    Note:
    The range specified in the prompt represents the total allowed over all ACDs. If you expend the entire allotment on ACD 1, you will have no more space available to other ACDs.

CAUTION:
You will lose external call record data if all the buffers on the CMS computer fill up. The buffers could become filled due to a link failure if the storage capacity of the receiving machine is exceeded or if the receiving machine goes down.

If you want to change the buffer size, you need to change the `Number of call records` field in the **`System Setup:`** `Data Storage Allocation` window. For more information, see the section on Data Storage Allocation in the appropriate CMS Administration for your CMS release.

Collection of external call records resumes when uucp finishes copying the files from the CMS buffers to the receiving computer.

Note:
Repeat Step 16 for each administered ACD.

The system displays the following message:

```
Start ECH in the on or off state: (default off)
```

17. Select whether ECH will start in the on or off state (default is off).
18. Verify that the installation completed successfully, enter:

    `tail /cms/install/logdir/admin.log`

    If the External Call History package is installed successfully, the system displays the following message:

```
External Call History package installed date/time
```

19. Enter:

    `cmsadm`

    The system displays the `Avaya Call Management System Administration Menu`.
20. Enter the number associated with the `run_cms` option.
21. Enter the number associated with the `Turn on CMS` option.

    The system displays the following message when CMS is on:

```
*** CMS is now up ***
```

Data collection with ECHI

When ECHI is off, records are still being collected in files, but the files are not sent to the receiving computer.

This section contains the following procedures:

Turning ECHI on or off

To turn ECHI on or off:

1. Enter:

   cmsadm

   The system displays the `Avaya Call Management System Administration Menu`.

2. Enter the number associated with the `run_pkg` option.

   The system displays a list of the installed CMS features.

3. Enter the number associated with the `external call history` option.

   The system displays the package status (on or off) and asks if you would like to turn it on or off:

```
ECHI is off. Do you wish to turn it on?
```

4. Enter **yes** or **no**, depending on what you want to do with the package. If you turn ECHI on, the system displays:

```
Do you wish to send the full call record buffer?
```

5. Enter **yes** or **no**, depending on what you want to do with the buffer.

CMS is not running and ECHI is off

If CMS is not running and if ECHI is off, the system displays the following message:

```
External Call History is off.
    Do you wish to turn it on? (default: yes)
```

- To turn ECHI on press **Enter**.

  The system displays the following message:

```
External Call History package turned on successfully
```

CMS is running and ECHI is off

If CMS is running and if ECHI is off, the system displays the following message:

```
External Call History is off.
    Do you wish to turn it on? (default: yes)
```

1. To turn ECHI on press **Enter**.

   The system displays the following message:

```
Do you wish to transmit the full buffer of call records?:
(default: yes)
```

2. Press **Enter**.

   The system displays the following message:

```
External Call History package turned on successfully
```

CMS is either running or not running and ECHI is on

If CMS is either running or not running and if ECHI is on, the system displays the following message:

```
External Call History is on.
    Do you wish to turn it off? (default: yes)
```

- To turn ECHI off press **Enter**.

  The system displays the following message:

```
External Call History package turned off successfully
```

Index