CHAPTER 4

Information Tags

Information tags (info-tags) are identifiers that can be used to retrieve information about call legs, events, the script itself, current configuration, and values returned from RADIUS.

Note Some info-tags have one or more parameters that are used to further identify the information to be retrieved, set, or modified.

Info-tags are grouped according to use. The first three characters of the info-tag label indicate the grouping:

- aaa—RADIUS information.
- cfg—Configuration information.
- con—Connection information.
- evt—Event information.
- leg—Call leg information.
- med—Media services information.
- sys—System information.

This chapter lists the available info-tags and the following information about each:

- Description—Explanation of the purpose of the info-tag.
- Syntax—The syntax of the info-tag.
- Mode—Whether the info-tag is read or read-write.
- Scope—The context in which the info-tag can be used. Some info-tags can be used at any time (global). Others are valid only when certain events are received, and the script terminates with error output if the info-tag is used in other situations. For example, you cannot call evt_dcdigits while handling the ev_setup_done event. In other words, if the previous command is **leg setup** and the ev_setup_done event has not yet returned, then you cannot execute an **infotag get evt_dcdigits** command, or the script will terminate with error output.
- Return Type—The type of information returned by the info-tag when used with an **infotag get** or **infotag set** command.
- Direct Mapping—Whether the info-tag can be used directly with a command (other than the **infotag get** or **infotag set** commands) and with which commands it can be used.

Note If an info-tag is specified incorrectly, if any of the parameters are specified incorrectly, or if the info-tag is used outside its intended scope, the script terminates with error output.

aaa_accounting_last_sent

Description	Retrieves the timestamp of the last accounting record sent from the voice-aaa subsystem.
Syntax	**infotag get aaa_accounting_last_sent** *{servertag}* • *servertag*—The server or server group identifier. This value refers to the method-list name, as in the following AAA configuration: `aaa accounting connection {default | method-list-name} group group-name`
Mode	Read
Scope	Valid only on completion of a successful servertag subscription.
Return Type	timestamp
Direct Mapping	None

aaa_avpair

Description	Returns the value of an AV-pair that was returned by RADIUS. After an **authorize** command finishes, the RADIUS server could have returned parameters as AV-pairs. This info-tag, along with aaa_avpair_exists, is used to get the value of a parameter after checking that such a parameter was returned. Refer to the table in "AV-Pair Names" section on page -4 for a list of valid VSA AV-pair names.
Syntax	**infotag get aaa_avpair** *avpair-name*
Mode	Read
Scope	Global
Return Type	String, Number, Boolean (1 or 0), or any other value that is configured or returned through RADIUS.
Direct Mapping	None

aaa_avpair_exists

Description	Returns the number of matched AV-pairs in the RADIUS server return. After an **authorize** command completes, the RADIUS server may return parameters as AV-pairs. This info-tag, along with aaa_avpair, is used to find out if a parameter exists before getting its value. Refer to the table in the "AV-Pair Names" section on page -4 for a list of valid VSA AV-pair names.
Syntax	**infotag get aaa_avpair_exists** *avpair-name*
Mode	Read
Scope	Global

Return Type	Number
Direct Mapping	None

AV-Pair Names

The info-tag aaa_avpair_exists can be used to check the availability of a VSA. The info-tag aaa_avpair can be used to access the value returned in this VSA. The valid VSA names that can be passed as parameters to these commands are the following.

Type	Name	Description
aaa	h323-ivr-in	A generic VSA for the billing server to send any information to the gateway in the form of an AV-pair, such as "color:blue" or "advprompt:rtsp://www.cisco.com/rtsp/areyouready.au"
	h323-ivr-out	A generic VSA for the gateway to send any information to the billing server in the form of an AV-pair, such as "color:blue" or "advprompt:rtsp://www.cisco.com/rtsp/areyouready.au"
	h323-credit-amount	The credit amount remaining in the account is returned.
	h323-credit-time	The credit time remaining in the account is returned.
	h323-prompt-id	The ID of the prompt is returned.
	h323-redirect-number	The number for redirection of a call is returned.
	h323-redirect-ip-addr	The IP address for the preferred route is returned.
	h323-preferred-lang	The language that the billing system returns as the preferred language of the end user. Three languages are supported; en (english), sp (spanish), and ch (mandarin). You can define additional languages as needed.
	h323-time-and-day	The time and day at the destination.
	h323-return-code	This information is returned only after an **authorization** command is issued. It returns either a numerical value or "Unknown variable name." The numerical value indicates what action the IVR application should take, namely to play a specific audio file to inform the end user of the reason for the failed authorization. If "Unknown variable name" is returned, the external AAA-server is out of service.
	h323-billing-model	Indicates the billing model used for the call. Initial values: 0=Credit, 1=Debit. Note: The debit card application assumes a Debit billing model.
	h323-currency	ISO currency to indicate what units to use in playing the remaining balance. The debit card application assumes units of *preferred_language*_dollar.au and *preferred_language*_cent.au.

Note If the *aaa* variable returns "0," this indicates that there is no VSA match to the name returned.

aaa_new_guid

Description	Request the system to generate and return a new GUID.
Syntax	**infotag get aaa_new_guid**
Mode	Read
Scope	Global
Return Type	String
Direct Mapping	None

cfg_avpair

Description	Returns the value of an AV-pair that was configured through the CLI.
Syntax	**infotag get cfg_avpair** *avpair-name*
Mode	Read
Scope	Global
Return Type	String, Number, Boolean (1 or 0), or any other value that is configured or returned through RADIUS.
Direct Mapping	None

cfg_avpair_exists

Description	Returns an indication of whether the specified parameter or AV-pair was configured through the CLI.
Syntax	**infotag get cfg_avpair_exists** *avpair-name*
Mode	Read
Scope	Global
Return Type	Boolean (1 = true; 0=false)
Direct Mapping	None

con_all

Description	Returns or maps to a list of all the connection IDs in the script.
Syntax	**infotag get con_all**
Mode	Read
Scope	Global
Return Type	Number list
Direct Mapping	Connections

con_ofleg

Description	Gets a list of all the connections the leg is a part of. This does not include those connections that are in Creation or under Destruction. The info-tag should map to just one leg.
Syntax	**infotag get con_ofleg** {*info-tag* | *legID*}
Mode	Read
Scope	Global
Return Type	Number list
Direct Mapping	Connections

evt_aaa_status_info

Description	Retrieves the aaa information. For example, the method list from the *ev_accounting_status_ind* event.
Syntax	**infotag get evt_aaa_status_info** *[attribute-name]* • *attribute-name*—The attribute you want to access. Defined attributes are *method-list-name* and *status*. If no attribute is specified, the list of attributes returns in the form of *av1#av2#*, where "#" is the delimiter. Status values are: – 000—Accounting Failed – 001—Accounting Success
Mode	Read
Scope	ev_accounting_status_ind
Return Type	String
Direct Mapping	None
Examples	set method_list [infotag get evt_aaa_status_info method-list-name]

evt_address_resolve_reject_reason

Description	Returns the address resolution rejection cause.
Syntax	**infotag get evt_address_resolve_reject_reason**
Mode	Read
Scope	ev_address_resolved
Return Type	Number
Direct Mapping	None

evt_address_resolve_term_cause

Description	Returns the address resolution termination cause.
Syntax	**infotag get evt_address_resolve_term_cause**
Mode	Read
Scope	ev_address_resolved
Return Type	Number
Direct Mapping	None

evt_connections

Description	Returns a list of connection IDs associated with the event received.
Syntax	**infotag get evt_connections**
Mode	Read
Scope	ev_handoff ev_returned ev_setup_done ev_create_done ev_destroy_done
Return Type	Number list
Direct Mapping	Connections

evt_consult_info

Description	Returns consult information from a consult response event.
Syntax	**infotag get evt_consult_info** {*consultID* | *transferDest*}
Mode	Read
Scope	ev_consult_response
Return Type	String
Direct Mapping	None

evt_dcdigits

Description	Returns the digits collected by the **leg collectdigits** command.
Syntax	**infotag get evt_dcdigits**
Mode	Read
Scope	ev_collectdigits_done
Return Type	String
Direct Mapping	None

evt_dest_handle

Description	Returns the application handle of the instance that registered for the destination number. This value is only available when the destination is an analog FXS phone and the phone is busy. If no application registered for the destination number, the value is an empty string.
Syntax	**infotag get evt_dest_handle**
Mode	Read
Scope	ev_setup_done
Return Type	String; represents a TCL application handle
Direct Mapping	None

evt_digit

Description	Returns the digit key that was pressed.
Syntax	**infotag get evt_digit**
Mode	Read
Scope	ev_digit_end
Return Type	String
Direct Mapping	None

evt_digit_duration

Description	Returns the duration of the digit that was pressed.
Syntax	**infotag get evt_digit_duration**
Mode	Read
Scope	ev_digit_end
Return Type	Number
Direct Mapping	None

evt_disc_iec

Description	Returns the Internal Error Code (IEC). In the case of multiple IECs, only the last one is returned. The IEC returns in the same format as Radius VSAs, as a dotted "version.entity.category.subsystem.error.diagnostic" string. If <*component*> is not specified, this command returns the dotted string with the components of the IEC. For example, "1.1.180.3.21.0." If there is no internal error associated with the disconnect, the returned string is "0.0.0.0.0.0."
Syntax	**infotag get evt_disc_iec** [<*component*>] The optional <*component*> argument can be one of the following parameters: • *entity*—Returns the entity component of the IEC. For example, "1." • *category*—Returns the category component of the IEC. For example, "180." • *subsystem—R*eturns the subsystem component. For example, "3." • *errcode*—Returns the subsystem-specific error code component. For example, "21."
Mode	Read
Scope	ev_disconnected
Return Type	String
Direct Mapping	None

evt_disc_rsi

Description	Returns the RSI numeric value that indicates where the release originated from. If there is no RSI available, this command returns 0. **Note** Valid only upon receiving an *ev_disconnected* event. If called while processing any other event, this command returns a TCL_ERROR, causing the script to terminate.
Syntax	**infotag get evt_disc_rsi**
Mode	Read
Scope	ev_disconnected
Return Type	String
Direct Mapping	None

evt_endpoint_addresses

Description	Returns a list of endpoint addresses.
Syntax	**infotag get evt_endpoint_addresses**
Mode	Read
Scope	ev_address_resolved
Return Type	String The return value has the following structure: *<endpointAddress>#<endpointAddress>#...* The first *endpointAddress* is the primary address. The *endpointAddresses* that follow are the alternate addresses.
Direct Mapping	None

evt_event

Description	Returns the name of the event received.
Syntax	**infotag get evt_event**
Mode	Read
Scope	Global
Return Type	String
Direct Mapping	None

evt_facility_id

Description	Returns the service type of the facility message response. The value is *ss_mcid_resp* for MCID invocation responses.
Syntax	**infotag get evt_facility_id**
Mode	Read
Scope	Global
Return Type	String
Direct Mapping	None
Example	set facility_id [infotag get evt_facility_id]
Usage Notes	None

evt_facility_report

Description	Enables the receipt of facility events.
Syntax	**infotag set evt_facility_report <***mcid* | *gtd***>**
Mode	Write
Scope	Global
Return Type	String
Direct Mapping	None
Example	infotag set evt_facility_report gtd
Usage Notes	• The *mcid* option of this information tag must be set to receive facility responses from MCID responses. • The *gtd* option of this information tag must be set to receive facility events that contain GTD information.

evt_feature_param

Description	Returns parameters related to a specific feature event.
Syntax	**infotag get evt_feature_param** *{parameter_name}* Possible return strings for media_inactivity are as follows: *no media received*—Media inactivity detected; no RTP or RTCP packets have been received for a configured amount of time. RTCP packet has been received before a media inactivity condition is met. *no control info received*—Media inactivity detected; no RTP or RTCP packets have been received for a configured amount of time. No RTCP packet has been received before a media inactivity condition is met.
Mode	Read
Scope	ev_feature
Return Type	String
Direct Mapping	None
Example	infotag get evt_feature_param media_inactivity_type

evt_feature_report

Description	Enables or disables certain feature events to be intercepted by the script.
Syntax	**infotag set evt_feature_report** *{["no_"]event_names}* • event_names—A list of application event names that define what events should or should not be reported to an application when a call is active. An event name with a *"no_"* prefix means not to report it. Possible values for event_names are as follows: – fax – modem – modem_phase – hookflash – onhook – offhook – media_inactivity
Mode	Write
Scope	ev_feature
Return Type	None
Direct Mapping	None
Examples	To enable hookflash and disable fax and modem feature events received by the script: `infotag set evt_feature_report hookflash nofax nomodem` To enable media_inactivity received by the script: `infotag set evt_feature_report media_activity`

evt_feature_type

Description	Returns the feature type string when a feature event is received.
Syntax	**infotag get evt_feature_type** • Possible event names returned are as follows: – fax – modem – modem_phase – hookflash – onhook – offhook – media_inactivity – media_activity
Mode	Read
Scope	ev_feature

Return Type	String
Direct Mapping	None

evt_gtd

Description	Associates a handle to the GTD parameters contained in the event. The application can use the handle to include the associated GTD parameters in any outgoing call signal message.
Syntax	**infotag get evt_gtd <gtd_handle>**
Mode	Read
Scope	ev_address_resolved ev_alert ev_connected ev_disconnected ev_proceeding ev_progress ev_setup_indication
Return Type	Number. If a handle can be created from the event, 1 is returned, otherwise 0 is returned.
Direct Mapping	None
Example	set handle [infotag get evt_gtd gtd_inf]
Usage Notes	None

evt_handoff ani

Description	Returns the ani set by the inbound application in the <transfer>/ field of the **leg setup** command.
Syntax	**infotag get evt_handoff ani**
Mode	Read
Scope	Global
Return Type	String
Direct Mapping	None
Example	`set ani [infotag get evt_handoff ani]`
Usage Notes	This command is only available in the handoff event.

evt_handoff argstring

Description	This command replaces the existing *evt_handoff_string* information tag.
Syntax	**infotag get evt_handoff argstring**
Mode	Read
Scope	Global
Return Type	String
Direct Mapping	None
Example	`set hstring [infotag get evt_handoff argstring]`
Usage Notes	This command is only available in the handoff event.

evt_handoff dnis

Description	Returns the dnis set by the inbound application in the <transfer>/ field of the **leg setup** command. Available only in the handoff event.
Syntax	**infotag get evt_handoff dnis**
Mode	Read
Scope	Global
Return Type	String
Direct Mapping	None
Example	`set dnis [infotag get evt_handoff dnis]`
Usage Notes	None

evt_handoff_legs

Description	Returns all the legs handed off to the application. Typically used to retrieve all call legs in a multiple-leg handoff, but can also be used for a single-leg handoff.
Syntax	**infotag get evt_handoff_legs**
Mode	Read
Scope	ev_handoff
Return Type	String; represents legIDs separated by space characters.
Direct Mapping	None

evt_handoff proto_headers

Description	Retrieves the handoff header.
Syntax	**infotag get evt_handoff proto_headers** [<*attribute-name*>] <*attribute-name*>—Name of the header to get.
Mode	Read
Scope	Global
Return Type	String
Direct Mapping	None

Example	`set AccountInfo [infotag get evt_handoff proto_headers AccountInfo]`
	The following command returns all headers in a concatenated string. Each header av-pair is separated by a '&':
	`set allHeaders [infotag get evt_handoff proto_headers]`

Usage Notes	If <attribute-name> is not specified, all headers are returned in a concatenated string, with each header separated by a "&" symbol.

evt_handoff_string

Description	Returns the handoff string when one or more call legs are handed off or returned to the script.
Syntax	**infotag get evt_handoff_string**
Mode	Read
Scope	ev_handoff ev_returned
Return Type	String
Direct Mapping	None

evt_iscommand_done

Description	Returns an indication of whether the command has finished.
Syntax	**infotag get evt_iscommand_done**
Mode	Read
Scope	ev_returned ev_setup_done ev_collectdigits_done ev_vxmldialog_done
Return Type	Boolean (1 = true; 0 = false)
Direct Mapping	None

evt_last_disconnect_cause

Description	Returns the value of the last failure detected during this call. The failure could have occurred on any call leg associated with this call. If no failures have occurred during the call, *di_000* is returned. The value of this information tag is updated while processing the following events: • ev_disconnected—Set to the cause value recieved in the protocol message. • ev_disc_prog_ind—Set to the cause value recieved in the protocol message. • ev_collectdigits_done—Set to *di_028* (invalid number) when the ev_collectdigits_done event returns status *cd_006*. Not modified when other digit collect status codes are returned. • ev_setup_done—Set to the cause code associated with the call setup attempt. The value is *di_016* (normal) if the call setup is successful. • ev_authenticate_done—Set to *di_057* (bearer capability is not available) when the ev_authenticate_done event status is not *au_000*. Not modified if event status is *au_000*. • ev_authorize_done—Set to *di_057* (bearer capability is not available) when the ev_authorize_done event status is not *ao_000*. Not modified if event status is *ao_000*.
Syntax	**infotag get evt_last_disconnect_cause**
Mode	Read
Scope	Global
Return Type	String. See the "Disconnect Cause" section on page 5-8 for string format.
Direct Mapping	None

evt_last_event_handle

Description	Returns the command handle of the setup.
Syntax	**infotag get evt_last_event_handle**
Mode	Read
Scope	ev_address_resolved ev_alert
Return Type	String
Direct Mapping	None

evt_last_iec

Description	When the script receives an *ev_setup_done* event that returns a bad status and wants more information, the script can poll this infotag to find the last Internal Error Code (IEC) associated with the rotary setup attempts and play an appropriate message to the caller on the incoming leg based on the IEC. This infotag is valid for any event. If there is no IEC associated with the call yet, the returned string is 0.0.0.0.0.0, or 0 for component.
Syntax	**infotag get evt_last_iec** [<*component*>] The optional <*component*> argument can be one of the following parameters: • *entity*—Returns the entity component of the IEC. For example, "1." • *category*—Returns the category component of the IEC. For example, "180." • *subsystem*—Returns the subsystem component. For example, "3." • *errcode*—Returns the subsystem-specific error code component. For example, "21."
Mode	Read
Scope	Global
Return Type	String
Direct Mapping	None

evt_legs

Description	Returns a list of leg IDs associated with the event received. For information about which legs the evt_legs info-tag returns for a specific event, see Chapter 5, "Events."
Syntax	**infotag get evt_legs**
Mode	Read
Scope	ev_authorize_done ev_leg_timer ev_digit_end ev_hookflash ev_disconnected ev_disconnect_done ev_grab ev_setup_indication ev_media_done ev_handoff ev_returned ev_setup_done ev_collectdigits_done ev_vxml_dialog_done ev_vxmldialog_event
Return Type	Number list
Direct Mapping	Legs

evt_module_handle

Description	Returns the module handle of the module that generated the current module event.
Syntax	**infotag get evt_module_handle**
Mode	Read
Scope	ev_module_done, ev_module_event
Return Type	String
Direct Mapping	None

evt_module_subevent

Description	Returns the module subevent name specified in the **module return interim** command that generated the *ev_module_event* being processed. This provides access to the parameters associated with the specific module event being handled.
Syntax	**infotag get evt_module_subevent**
Mode	Read
Scope	ev_module_event
Return Type	String
Direct Mapping	None

evt_module_context

Description	Returns the module context specified when creating this module with the **module new** command.
Syntax	**infotag get evt_module_context**
Mode	Read
Scope	ev_module_done, ev_module_event
Return Type	String
Direct Mapping	None

evt_msg

Description	Retrieves the message body. **Note** Only valid when handling an ev_msg_indication event.
Syntax	**infotag get evt_msg** <*array-name*> <array_name>—The name of a TCL array populated with information from the -p attribute in the sendmsg command.
Mode	Read
Scope	Global
Return Type	A list of array item names.

Direct Mapping	None

Examples	`proc act_handle_msg {`
	`  infotag get` **`evt_msg`** `rx_message`
	`  if { [info exists rx_message(fruit)] == 1 } {`
	`  set fruit $rx_message(fruit)`
	`  } else { set fruit “peaches” }`
	`  set ident [infotag get evt_msg_source]`
	`  set return_msg(text) “Got your message requesting $fruit, ignoring it”`
	`  sendmsg $ident -p return_msg`
	`}`

Usage Notes	None

evt_msg_source

Description	Retrieves the handle of the source of the message.
Syntax	**infotag get evt_msg_source**
Mode	Read
Scope	Global
Return Type	String
Direct Mapping	None

Examples	`proc act_handle_msg {`
	`  infotag get evt_msg rx_message`
	`  if { [info exists rx_message(fruit)] == 1 } {`
	`  set fruit $rx_message(fruit)`
	`  } else { set fruit "peaches" }`
	`  set ident [infotag get` **`evt_msg_source`**`]`
	`  set return_msg(text) "Got your message requesting $fruit, ignoring it"`
	`  sendmsg $ident -p return_msg`
	`}`

Usage Notes	• This handle is the same kind of handle returned by the *mod_handle* information tag. It can be used in a **sendmsg** or **handoff** command. • This information tag is only valid after an ev_msg_indication event. If called at other times, an error occurs and the script fails.

evt_params

Description	Creates an array variable named <*array-name*> within the current modulespace from the information provided by the <*param-array*> parameter of the **module return** command. This provides access to the parameters associated with the specific module event being handled.
Syntax	**infotag get evt_params** <*array-name*>
Mode	Read
Scope	ev_module_done, ev_module_event
Return Type	String
Direct Mapping	None

evt_progress_indication

Description	Returns the value of the progress indication of the received alert, connected, disconnect, disconnect with PI, proceeding, or progress message.
Syntax	**infotag get evt_progress_indication**
Mode	Read

Scope	ev_alert ev_connected ev_progress ev_proceeding ev_disconnected ev_disc_prog_ind
Return Type	Number
Direct Mapping	None
Example	set progress [infotag get evt_progress_indication]
Usage Notes	None

evt_proto_content

Description	Used to retrieve the content of the received event. The content is the body of the protocol message.
Syntax	**infotag get evt_proto_content**
Mode	Read
Scope	ev_notify, ev_subscribe_done, ev_unsubscribe_indication
Return Type	String
Direct Mapping	None
Examples	`set r_content [infotag get evt_proto_content]`
Usage Notes	Only textual content, such as plain text or xml text, is supported.

evt_proto_content_type

Description	Used to retrieve the content type of the received event. This event specifies the type of content carried in the body of the protocol message.
Syntax	**infotag get evt_proto_content_type**
Mode	Read
Scope	ev_notify, ev_subscribe_done, ev_unsubscribe_indication
Return Type	String
Direct Mapping	None
Examples	`set content_type [infotag get evt_proto_content_type]`
Usage Notes	Only textual content, such as plain text or xml text, is supported.

evt_proto_headers

Description	Used to access the protocol header information associated with the events.
Syntax	**infotag get evt_proto_headers** <*attribute-name*> <*attribute-name*>—Name of the header to get.
Mode	Read
Scope	ev_notify, ev_subscribe_done, ev_unsubscribe_indication
Return Type	String containing the value of the header attribute.
Direct Mapping	None

Examples	`set event [infotag get evt_proto_headers Event]`
	`set inviteSubject [infotag get evt_proto_headers "Subject"]`
	`set all_headers [infotag get evt_proto_headers]`

Usage Notes	• Both standard and non-standard headers can be accessed using this command. The application does not cache the header values. If an application wants to retain header information, it must save the information in its local or global variables. • If <attribute-name> is not specified, all headers are returned in a concatenated string, with each header separated by a "&" symbol.

evt_report ev_transfer_request

Description	Allows notification of the call transfer request event from an endpoint to the application.
Syntax	**infotag set evt_report ev_transfer_request**
Mode	Write
Scope	Global
Return Type	String
Direct Mapping	None
Examples	`infotag set evt_report ev_transfer_request`
Usage Notes	• The script performs a leg setup to the transfer-to endpoint. The callinfo (transferCall, transferBy, consultID) field is populated with information available in the evt_transfer_info information tag. • After the callinfo field is populated, a call is set up toward the transfer-to endpoint.

evt_redirect_info

Description	Returns forwarding request information when a call is being forwarded.
Syntax	**infotag get evt_redirect_info** {*redirectDest* | *redirectReason* | *redirectCount* | *originalDest*} • redirectDest—redirected-to number retrieved during call setup to the destination • redirectReason—the type of redirection – rr_cfb—CF-busy – rr_cfnr—CF-no answer – rr_cd—CD-call deflection – rr_cfu—CF-unconditional • redirectCount—number of call diversions that have occurred • originalDest—original called number
Mode	Read
Scope	ev_setup_done
Return Type	String
Direct Mapping	None

evt_service_control

Description	Returns the service control indexed by <*index*>, with <*index*> *1* being the first service control field.
Syntax	**infotag get evt_service_control <index>**
Mode	Read
Scope	ev_address_resolved
Return Type	String The string content is application dependent. The format of the content are agreed upon between the application and the route entity. **Note** The application processes the service descriptor fields. Neither the gatekeeper nor the gateway interprets the contents of the service descriptors.
Direct Mapping	None

evt_service_control_count

Description	Returns the number of service control fields.
Syntax	**infotag get evt_service_control_count**
Mode	Read
Scope	ev_address_resolved
Return Type	Number
Direct Mapping	None

evt_status

Description	Returns the status of the event received. This info-tag is valid only in the scope of the function handling the event. For a list of possible statuses, see the "Status Codes" section on page 5-6.
Syntax	**infotag get evt_status**
Mode	Read
Scope	ev_setup_done ev_collectdigits_done ev_media_done ev_disconnected ev_authorize_done ev_authenticate_done ev_vxmldialog_done ev_subscribe_done ev_unsubscribe_done ev_unsubscribe_indication
Return Type	String
Direct Mapping	None

Examples	`proc act_SubscribeDone { } {`
	`  puts "***** act_Subscribe : SUBSCRIPTION DONE received"`
	`  set sub_id [infotag get evt_subscription_id]`
	`  set status [infotag get evt_status]`
	`  puts "*** act_subscribe: subscription status=$status"`
	`  if {$status == "su_000"} {`
	`  puts "\n Subscription accepted."`
	`  return`
	`  } else if {$status == "su_001"} {`
	`  puts "\n Subscription is pending"`
	`  return`
	`  } else {$status == "su_002"} {`
	`  # subscription request has failed; close the subscription`
	`  puts "\nSubscription request has failed."`
	`  set status_text [infotag get evt_status_text]`
	`  puts "text is: $status_text\n"`
	`  subscription close $sub_id`
	`  }`
	`}`

Usage Notes	Return codes indicate some type of failure has occurred; therefore, the script should be written to handle such errors accordingly.

evt_status_text

Description	Retrieves failure information associated with an event.
Syntax	**infotag get evt_status_text**
Mode	Read
Scope	ev_subscribe_done, ev_unsubscribe_done, ev_unsubscribe_indication
Return Type	String with failure information, if any.
Direct Mapping	None

Example	`set sub_id [infotag get evt_subscription_id]`
	`set status [infotag get evt_status]`
	`puts "*** act_subscribe: subscription status=$status"`
	`  if {$status == "su_002"} {`
	`  # subscription request has failed; close the subscription`
	`  puts "\nSubscription request has failed."`
	`  set status_text [infotag get evt_status_text]`
	`  puts "text is: $status_text\n"`
	`  subscription close $sub_id`
	`  }`

evt_subscription_id

Description	Retrieves the subscription id associated with events related to a subscription or a notification.
Syntax	**infotag get evt_subscription_id**
Mode	Read
Scope	ev_subscribe_done, ev_notify, ev_subscribe_cleanup, ev_unsubscribe_indication
Return Type	Subscription ID
Direct Mapping	None
Examples	`set sub_id [infotag get evt_subscription id]`
Usage Notes	None

evt_timer_name

Description	Retrieves the name associated with the expired named_timer.
Syntax	**infotag get evt_timer_name**
Mode	Read
Scope	ev_named_timer
Return Type	String
Direct Mapping	None

evt_transfer_info

Description	Returns transfer information from a transfer request event.
Syntax	**infotag get evt_transfer_info** {*transferBy* | *transferDest* | *consultID*}
Mode	Read
Scope	ev_transfer_request
Return Type	String
Direct Mapping	None

evt_vxmlevent

Description	Returns a string containing the VXML event that was thrown. These events are generally of the form *vxml.**. Events thrown from the dialog markup, or the document using the VXML *sendevent object* extension, are of the form vxml.dialog.*. For more information on sendevent objects, refer to SendEvent Object, page 1-8. Events thrown by the system due to some event, such as the vxml document executing a <disconnect/> tag, are of the form vxml.session.*.
Syntax	**infotag get evt_vxmlevent**
Mode	Read
Scope	ev_vxmldialog_done ev_vxmldialog_event
Return Type	String
Direct Mapping	None

evt_vxmlevent_params

Description	Retrieves parameters that may come with an event. This info-tag clears the array variable and fills it with the parameter values indexed by the parameter names in the param option of the sendevent object tag. Parameters can also be returned through the <exit/> tag with a namelist attribute. For more information on sendevent objects, refer to SendEvent Object, page 1-8. In either case, if the namelist contains an audio clip variable, it is made available to the Tcl script as a parameter with a string value containing the *ram:// uri* form for the audio clip. The info tag returns a space-separated list of indexes that were added to the return array variable passed as a parameter to the information tag.
Syntax	**infotag get evt_vxmlevent_params** <*array-variable-name*>
Mode	Read
Scope	ev_vxmldialog_done ev_vxmldialog_event
Return Type	String Parameter: array-variable-name
Direct Mapping	None

gtd_attr_exists

Description	Used to determine if an attribute instance exists in a GTD message.
Syntax	**infotag get gtd_attr_exists** <*gtd_handle*><*attr_instance*> • <*gtd_handle*>—Name of the GTD handle from which the application wants to check the existence of a GTD attribute instance. • <*attr_instance*>—This parameter is of the form <*attr_name*>, <*attr_instance*>. <*attr_instance*> can be specified with a value of -1, which means "don't care."
Mode	Read
Scope	Global
Return Type	Boolean (1=true; 0=false)
Direct Mapping	None

last_command_handle

Description	Retrieves the last command handle.
Syntax	**infotag get last_command_handle**
Mode	Read
Scope	Global
Return Type	String
Direct Mapping	None

leg_all

Description	Returns or maps to one or more call legs. This is the union of leg_incoming and leg_outgoing.
Syntax	**infotag get leg_all**
Mode	Read
Scope	Global
Return Type	Number list
Direct Mapping	Legs

leg_ani

Description	Returns the ANI field of CallInfo.
Syntax	**infotag get leg_ani** [*legID*] If no leg ID is specified, this info-tag returns the ANI field of the first incoming call leg. Not specifying a leg ID works only if there is at least one incoming call leg. If a leg ID is specified, this info-tag returns the ANI field of that call leg. If the call leg is not valid, the script terminates with error output.
Mode	Read
Scope	Global
Return Type	String
Direct Mapping	None

leg_ani_pi

Description	Gets the calling number presentation indication value.
Syntax	**infotag get leg_ani_pi**
Mode	Read
Scope	Global
Return Type	Number list Values retrieved could be one of the following: 1—presentation_allowed 2—presentation_restricted 3—number_lost_due_to_interworking 4—reserved_value 5—not_present (denotes that the Calling Number IE is absent in the incoming signaling message.
Direct Mapping	None

leg_ani_si

Description	Gets the calling number screening indication value.
Syntax	**infotag get leg_ani_si**
Mode	Read
Scope	Global
Return Type	Number list Values retrieved could be one of the following: 1—usr_provided_unscreened 2—usr_provided_screening_passed 3—usr_provided_screening_failed 4—network_provided 5—not_present (denotes that the Calling Number IE is absent in the incoming signaling message.
Direct Mapping	None

leg_dn_tag

Description	Returns the DN field of call info. In an Ephone-initiated call, it carries the DN tag of the calling party.
Syntax	**infotag get leg_dn_tag** *legID*
Mode	Read
Scope	Global
Return Type	String
Direct Mapping	None

leg_dnis

Description	Returns the DNIS field of CallInfo.
Syntax	**infotag get leg_dnis** [*legID*] If no leg ID is specified, this info-tag returns the DNIS field of the first incoming call leg. Not specifying a leg ID works only if there is at least one incoming call leg. If a leg ID is specified, this info-tag returns the DNIS field of that call leg. If the call leg is not valid, the script terminates with error output.
Mode	Read
Scope	Global
Return Type	String
Direct Mapping	None

leg_display_info

Description	Returns the display_info field of call info. In an Ephone-initiated call, this field contains the name of the calling party.
Syntax	**infotag get leg_display_info** *legID*
Mode	Read
Scope	Global
Return Type	String
Direct Mapping	None

leg_guid

Description	Returns the GUID o a leg.
Syntax	**infotag get leg_guid** [*legID*] If legID is not specified, returns the GUID of the first incoming leg.
Mode	Read
Scope	Global
Return Type	String
Direct Mapping	None

leg_incoming

Description	Returns or maps to one or more incoming call legs.
Syntax	**infotag get leg_incoming**
Mode	Read
Scope	Global
Return Type	Number list
Direct Mapping	Legs

leg_incoming_guid

Description	Returns the incoming GUID of a leg.
Syntax	**infotag get leg_incoming_guid** [legID] If legID is not specified, returns the GUID of the first incoming leg.
Mode	Read
Scope	Global
Return Type	String
Direct Mapping	None

leg_inconnection

Description	Gets a list of legs that are part of this connection. The info-tag parameter maps to just one connection.
Syntax	**infotag get inconnection** {*connID* | *info-tag*}
Mode	Read
Scope	Global
Return Type	Number list
Direct Mapping	Legs

leg_isdid

Description	Returns the DID field of CallInfo. This is a Boolean field (1 and 0) that reflects the FinalDestination flag of the call leg.
Syntax	**infotag get leg_isdid** [*legID*] If no leg ID is specified, this info-tag returns the DID field of the first incoming call leg. Not specifying a leg ID works only if there is at least one incoming call leg. If a leg ID is specified, this info-tag returns the DID field of that call leg. If the call leg is not valid, the script terminates with error output.
Mode	Read
Scope	Global
Return Type	Boolean (1 = true; 0 = false)
Direct Mapping	None

leg_outgoing

Description	Returns or maps to one or more outgoing call legs.
Syntax	**infotag get leg_outgoing**
Mode	Read
Scope	Global
Return Type	Number list
Direct Mapping	Legs

leg_password

Description	If no leg ID is specified, this info-tag returns the password field of the first incoming call leg. Not specifying a leg ID works only if there is at least one incoming call leg. If a leg ID is specified, this info-tag returns the password field of that call leg. If the call leg is not valid, the script terminates with error output.
Syntax	**infotag get leg_password** [*legID*]
Mode	Read
Scope	Global
Return Type	String
Direct Mapping	None

leg_proto_headers

Description	Provides access to headers associated with the protocol being used.
Syntax	**infotag get leg_proto_headers** [<*attribute-name*>] [*legID*] <*attribute-name*>—Name of the header to get.
Mode	Read
Scope	Global
Return Type	String
Direct Mapping	None

Example	`set AccountInfo [infotag get leg_proto_headers AccountInfo]`
	`set inviteSubject [infotag get evt_proto_headers "Subject"]`
	`set inviteFrom [infotag get leg_proto_headers "From" leg_incoming]`
	The following command returns all headers received from the incoming Invite message in a concatenated string. Each header av-pair is separated by a '&':
	`set allHeaders [infotag get evt_proto_headers]`

Usage Notes	• This information tag allows the accessing of SIP headers from VXML documents or TCL IVR 2.0 scripts. • Currently, only access to headers in SIP invite, subscribe, notify and H.323 setup messages are supported. • If <attribute-name> is not specified, all headers are returned in a concatenated string, with each header separated by a "&" symbol. • If legID is not provided, the first incoming leg is applied.

leg_rdn

Description	Gets the redirect number from the first incoming leg.
Syntax	**infotag get leg_rdn**
Mode	Read
Scope	Global
Return Type	String
Direct Mapping	None

leg_rdn_pi

Description	Gets the redirect number presentation indication value.
Syntax	**infotag get leg_rdn_pi**
Mode	Read
Scope	Global
Return Type	Number list Values retrieved could be one of the following: 1—presentation_allowed 2—presentation_restricted 3—number_lost_due_to_interworking 4—reserved_value 5—not_present (denotes that the Redirect Number IE is absent in the incoming signaling message.
Direct Mapping	None

leg_rdn_si

Description	Gets the redirect number screening indication value.
Syntax	**infotag get leg_rdn_si**
Mode	Read
Scope	Global
Return Type	Number list Values retrieved could be one of the following: 1—usr_provided_unscreened 2—usr_provided_screening_passed 3—usr_provided_screening_failed 4—network_provided 5—not_present (denotes that the Redirect Number IE is absent in the incoming signaling message.
Direct Mapping	None

leg_redirect_cnt

Description	Retrieves redirection count information from the first incoming call leg or for a leg if callid is specified.
Syntax	**infotag get leg_redirect_cnt**
Mode	Read
Scope	Global
Return Type	Number. Values retrieved between 0–7.
Direct Mapping	None

leg_remoteipaddress

Description	Returns the remote IP address of the endpoint from which the call is received. If the IP address is not available, an empty string is returned.
Syntax	**infotag get leg_remoteipaddress** <leg-id>
Mode	Read
Scope	Global
Return Type	String (ip address)
Direct Mapping	None

leg_remote_media_ip_address

Description	Returns the remote media IP address of the endpoint. If the IP address is not available, an empty string is returned.
Syntax	**infotag get leg_remote_media_ip_address** <leg_id>
Mode	Read
Scope	Global
Return Type	String (ip address)
Direct Mapping	None

leg_remote_signaling_ip_address

Description	Returns the remote signaling IP address of the endpoint. If the IP address is not available, an empty string is returned.
Syntax	**infotag get leg_remote_signaling_ip_address** <leg_id>
Mode	Read
Scope	Global
Return Type	String (ip address)
Direct Mapping	None

leg_rgn_noa

Description	Gets the redirect number nature of address value.
Syntax	**infotag get leg_rgn_noa**
Mode	Read
Scope	Global
Return Type	Number Values retrieved could be one of the following: 00—Unknown, number present 01—Unknown, number absent, presentation restricted 02—Unique subscriber number 03—Nonunique subscriber number 04—Unique national (significant) number 05—Nonunique national number 06—Unique international number 07—Nonunique international number 08—Network specific number 09—Nonsubscriber number 10—Subscriber number, operator requested 11—National number, operator requested 12—International number, operator requested 13—No number present, operator requested 14—No number present, cut through call to carrier 15—950+ call from local exchange carrier public station, hotel/motel or non-exchange access end office 16—Test line test code 17—Unique 3 digit national number 18—Credit card 19—International inbound 20—National or international with carrier access code included 21—Cellular - global ID GSM 22—Cellular - global ID NWT 900 23—Cellular - global ID autonet 24—Mobile (other) 25—Ported number 26—VNET 27—International operator to operator outside WZ1 28—International operator to operator inside WZ1 29—Operator requested - treated 30—Network routing number in national (significant) format 31—Network routing number in network specific format 32—Network routing number concatenated with called directory number 33—Screened for number portability 34—Abbreviated number
Direct Mapping	None

Note This infotag has been provided as an interim mechanism for accessing specific signaling information. It may be obsoleted in a future IOS release when an alternate method of accessing this information is made

available.

leg_rgn_npi

Description	Returns the redirect number numbering plan indicator value.
Syntax	**infotag get leg_rgn_npi**
Mode	Read
Scope	Global
Return Type	Number Values retrieved could be one of the following: 1—ISDN numbering plan 2—Data numbering plan 3—Telex numbering plan 4—Private numbering plan 5—National 6—Maritime mobile 7—Land mobile 8—ISDN mobile 252—Unknown
Direct Mapping	None

Note This infotag has been provided as an interim mechanism for accessing specific signaling information. It may be obsoleted in a future IOS release when an alternate method of accessing this information is made available.

leg_rgn_num

Description	Returns the redirect number address.
Syntax	**infotag get leg_rgn_num**
Mode	Read
Scope	Global
Return Type	String
Direct Mapping	None

Note This infotag has been provided as an interim mechanism for accessing specific signaling information. It may be obsoleted in a future IOS release when an alternate method of accessing this information is made available.

leg_rgn_pi

Description	Returns the redirect number presentation indicator value.
Syntax	**infotag get leg_rgn_pi**
Mode	Read
Scope	Global
Return Type	Number Values retrieved could be one of the following: 0—Unknown 1—Presentation allowed 2—Presentation not allowed 3—Address not available
Direct Mapping	None

Note This infotag has been provided as an interim mechanism for accessing specific signaling information. It may be obsoleted in a future IOS release when an alternate method of accessing this information is made available.

leg_rgn_si

Description	Returns the redirect number screening indicator value.
Syntax	**infotag get leg_rgn_si**
Mode	Read
Scope	Global
Return Type	Number Values retrieved could be one of the following: 1—User provided not screened 2—User provided screening passed 3—User provided screening failed 4—Network provided 252—Unknown
Direct Mapping	None

Note This infotag has been provided as an interim mechanism for accessing specific signaling information. It may be obsoleted in a future IOS release when an alternate method of accessing this information is made available.

leg_settlement_time

Description	Returns the minimum of the OSP settlement time (in seconds) associated with the list of specified legs.
Syntax	**infotag get leg_settlement_time** {**legID** | **info-tag**} [**minimum**] If you specify minimum, this returns the minimum of the OSP settlement time of the list of legs and the value of the AAA AV-pair creditTime. This AAA AV-pair creditTime was returned by a previous **aaa authorize** command. If all credit times are uninitialized, "uninitialized" is returned. If all have unlimited time, or if one is uninitialized and the others have unlimited time, "unlimited" is returned.
Mode	Read
Scope	Global
Return Type	Number
Direct Mapping	None

leg_source_carrier_id

Description	Retrieve the source carrier ID.
Syntax	**infotag get leg_source_carrier_id**
Mode	Read
Scope	Global
Return Type	None
Direct Mapping	None

leg_state

Description	Returns the state of the call leg in a different state.
Syntax	**infotag get leg_state**
Mode	Read
Scope	Global
Return Type	None
Direct Mapping	None

leg_subscriber_type

Description	Returns the subscriber type.
Syntax	**infotag get leg_subscriber_type**
Mode	Read
Scope	Global
Return Type	None
Direct Mapping	None

leg_suppress_outgoing_auto_acct

Description	When set, the service provider module does not automatically generate an accounting packet on the outgoing call leg.
Syntax	**infotag get leg_suppress_outgoing_auto_acct** **infotag set leg_suppress_outgoing_auto_acct**
Mode	Read/write
Scope	Global
Return Type	None for set Boolean (0 | 1) for get
Direct Mapping	Leg

leg_target_carrier_id

Description	Set the target carrier ID.
Syntax	**infotag set leg_target_carrier_id** **Note** This command is obsolete and is replaced by the **callinfo** *targetCarrierID* command.
Mode	Write
Scope	Global
Return Type	String
Direct Mapping	None

leg_tdm_hairpin

Description	Enables TDM hairpinning on the incoming call leg. Copies the TDM hairpin value to the outgoing leg and enables it for TDM hairpinning.
Syntax	**infotag set leg_tdm_hairpin** [legID | info-tag] [enable | disable]
Mode	Write
Scope	Global
Return Type	None
Direct Mapping	None

leg_type

Description	Returns the interface type of the specified leg. If the *detail* parameter is specified, this returns a detailed string that includes the interface subtype of the specified leg. The *detail* parameter is optional. The information tag used to specify the leg should map to just one leg.
Syntax	**infotag get leg_type** [detail] {legID | info-tag}
Mode	Read
Scope	Global

Return Type	String Possible return values are: telephony voip mmoip voatm vofr unknown none Possible return values with the *detail* parameter are: voip-h323 voip-sip tele-analog-em tele-analog-fxo tele-analog-fxs tele-analog-efxs tele-analog-efxo tele-digital-isdn tele-digital-cas tele-digital-bri tele-digital-r2 msp-doc msp-fax unknown
Direct Mapping	None

leg_username

Description	If no leg ID is specified, this info-tag returns the username field of the first incoming call leg. Not specifying a leg ID works only if there is at least one incoming call leg. If a leg ID is specified, this info-tag returns the username field of that call leg. If the call leg is not valid, the script terminates with error output.
Syntax	**infotag get leg_username** [*legID*]
Mode	Read
Scope	Global
Return Type	String
Direct Mapping	None

med_backup_server

Description	Returns or sets the backup server. This is applicable for RTSP-based prompts. If the script attempts to play a prompt using a URL and the URL fails, it tries to replay the URL from a list of backup servers by replacing the server portion of the URL. For example, if the script tries (but fails) to play a prompt from: rtsp://www.cisco.com:5554/audiofiles/english/anounce.au and the backup server 0 is configured as: rtsp://www.real.com/cisco/ then the backup URL attempted is: rtsp://www.real.com/cisco/audiofiles/english/anounce.au A maximum of two (0 and 1) backup servers can be configured. This info-tag applies only to streams on which you have not played any prompts and is typically used in the one-time initialization section of the script.
Syntax	**infotag get med_backup_server** *index* **infotag set med_backup_server** *index server-URL*
Mode	Read/Write
Scope	Global
Return Type	String
Direct Mapping	None

med_language

Description	Returns or sets the current active language for media playout. This info-tag returns the language index or the language prefix (depending on whether prefix is specified) for the currently active language.
Syntax	**infotag get med_language** [**prefix**] **infotag set med_language** [*index* | **prefix** *prefix*]
Mode	Read/Write
Scope	Global
Return Type	String/Number
Direct Mapping	None

med_language_map

Description	Returns or sets the mapping between the language index and the language prefix. This info-tag returns the language index or the language prefix (depending on whether prefix is specified) for the currently active language.
Syntax	**infotag get med_language_map** [*index* | **prefix** *prefix*] **infotag set med_language_map** *index prefix*
Mode	Read/Write
Scope	Global
Return Type	String/Number
Direct Mapping	None

med_location

Description	Returns or sets the language locations for all the languages the script uses. The language prefix, category, and location are the same as those configurable from the Cisco IOS command line interface (CLI).
Syntax	**infotag get med_location** *prefix category.* Valid category values are 1, 2, 3, 4. **infotag set med_location** *prefix category location.* Category 0 can be used to set all 1–4 categories.
Mode	Read/Write
Scope	Global
Return Type	String
Direct Mapping	None

med_total_languages

Description	Returns the total number of languages configured.
Syntax	**infotag get med_total_languages**
Mode	Read
Scope	Global
Return Type	Number
Direct Mapping	None

media_timer_factor

Description	Sets the receive-rtcp timer. The new value is used within the scope of the script and does not change the gateway configuration.
Syntax	**infotag set media_timer_factor** <*timer_factor*> *timer_factor*—An integer between 2–1000. The value is a multiple of the RTCP transmission interval. A value of 5 is recommended. **Note** If a value outside the range of 2–1000 is used, the script receives an error message.
Mode	Write
Scope	None
Return Type	None
Direct Mapping	None
Example	infotag set media_timer_factor 5

mod_all_handles

Description	Retrieves a list of all instances running on the gateway. The returned handle can be used as an argument for **handoff** and **sendmsg** commands.
Syntax	**infotag get mod_all_handles**
Mode	Read
Scope	Global
Return Type	String
Direct Mapping	None
Examples	`set all_handles [infotag get mod_all_handles]`
Usage Notes	• Handles are ASCII strings that are only valid within a gateway. A handle for a session on another gateway cannot be used to send a message to that gateway. • The format of a handle is not designed to be parsed or printed by a TCL script. The handle is used internally by a sendmsg or handoff command.

mod_handle

Description	Retrieves the handle of the currently running application session. The returned handle can be used as an argument for **handoff** and **sendmsg** commands.
Syntax	**infotag get mod_handle**
Mode	Read
Scope	Global
Return Type	String
Direct Mapping	None

Examples	`set id [infotag get mod_handle]`
	
	`set my_name new_customer`
	`set recorded_name ([infotag get mod_handle]) $my_name`

Usage Notes	• Handles are ASCII strings that are only valid within a gateway. A handle for a session on another gateway cannot be used to send a message to that gateway. • The format of a handle is not designed to be parsed or printed by a TCL script. The handle is used internally by a **sendmsg** or **handoff** command. • Returns “unavailable” if the service is not running.

mod_handle_service

Description	Retrieves the handle of the named service.
Syntax	**infotag get mod_handle_service** <*service*>
Mode	Read
Scope	Global
Return Type	String
Direct Mapping	None
Example	`set serv_hndl [infotag get mod_handle_service]`
Usage Notes	• Returns “unavailable” if the named service is not running. • The returned handle can be used as an argument for **handoff** and **sendmsg** commands.

set iec

Description	Sets the Internal Error Code (IEC) before issuing a **call close** or **leg disconnect** command. **Note** Using the *set iec* information tag after specifying the IEC with the **leg disconnect** -<*iec*> command causes duplicate IECs to be associated with the call leg.
Syntax	**infotag set iec** <*legID* | *info-tag*> <*iec*> <*iec*>—Can be one of the following arguments: • *media_done_err*—Indicates that the script is terminating the call because of an error status returned by the ev_media_done event. • *collectdigits_done_err*—Indicates that the script is terminating the call because of an error status returned by the ev_collectdigits_done event. • *authenticate_done_err*—Indicates that the script is terminating the call because of an error status returned by the ev_authenticate_done_event. • *authorize_done_err*—Indicates that the script is terminating the call because of an error status returned by the ev_authorize_done event. • *media_inactivity_err*—Disconnects the call when media inactivity is detected and reported. • *accounting_conn-err*—Indicates the script detects that connectivity to the accounting server is lost.
Mode	Write
Scope	Global
Return Type	String
Direct Mapping	None

subscription_context

Description	Retrieves the subscription context information associated with the subscription ID.
Syntax	**infotag get subscription_context** <*subscription_id*> *[attribute]* • *subscription_id*—ID associated with the subscription • *attribute*—The attribute within the context information.
Mode	Read
Scope	Global
Return Type	String containing the context.
Direct Mapping	None
Example	`set accountNumber [infotag get subscription_context $MySubID accountNumber]`
Usage Notes	If the *attribute* argument is missing, a string containing all attributes and values is returned. Each av-pair is escaped by the "#" character.

subscription_info

Description	Retrieves the subscription information associated with a subscription ID. Retrievable subscription attributes are url, event, expirationTime, subscription_context, and notificationReceiver.
Syntax	**infotag get subscription_info** <*subscription_id*> <*attribute*> • *subscription_id*—ID associated with the subscription • *attribute*—The attribute to access.
Mode	Read
Scope	Global
Return Type	String
Direct Mapping	None
Example	`set exp_time [infotag get subscription_info $sub_id expirationTime]`
Usage Notes	None

subscription_server_ipaddress

Description	Retrieves the IP address of the subscription server.
Syntax	**infotag get subscription_server_ipaddress** <*subscription_id*> • *subscription_id*—ID associated with the subscription
Mode	Read
Scope	Global
Return Type	String containing the IP address of the server.
Direct Mapping	None
Example	`set ipaddr [infotag get subscription_server_ipaddress $sub_id]`
Usage Notes	None

sys_version

Description	Returns the version of the Tcl IVR API.
Syntax	**infotag get sys_version**
Mode	Read
Scope	Global
Return Type	String
Direct Mapping	None