Features Supported by TSP

This chapter describes the features that Cisco Unified TAPI Service Provider (TSP) supports. See New and Changed Information for described features, which are listed by Unified Communications Manager release.

3XX

Cisco TSP maps the CTI reason code for 3XX to REDIRECT. When a call arrives on a monitored line due to 3XX feature, the call reason for the incoming call will get REDIRECT in this case. No interface change for TSP 3XX support.

Backward Compatibility

This feature is backward compatible.

Additional Features Supported on SIP Phones

Unified Communications Manager extends support for phones that are running SIP with these new features:

- PhoneSetLamp (but only for setting the MWI lamp)
- PhoneSetDisplay
- PhoneDevSpecific (CPDST_SET_DEVICE_UNICODE_DISPLAY)
- LineGenerateTone
- Park and UnPark
- The LINECALLREASON_REMINDER reason for CallPark reminder calls

- PhoneGetDisplay (but only after a PhoneSetDisplay)

TSP does not pass unicode name for phones that are running SIP.

AES 256 Algorithm IDs

From release 10.5(2) Cisco Unified Communications Manager has updated its list of supported encryption algorithm IDs. CiscoTSP has also updated its list of SRTP AlgorithmIDs. The new CiscoSRTPAlgorithm IDs added to CiscoLineDevSpecificMsg.h are:

- SRTP_AES_CM_128_HMAC_SHA1_80
- SRTP_F8_128_HMAC_SHA1_32
- SRTP_F8_128_HMAC_SHA1_80
- SRTP_AEAD_AES_128_GCM
- SRTP_AEAD_AES_256_GCM

The SRTP_AEAD_128_GCM and SRTP_AEAD_AES_256_GCM will be negotiated only for secure calls between two SIP endpoints.

CTI Ports can register with the above mentioned algorithm IDs, but will negotiate only on the existing SRTP_AEAD_128_COUNTER for a secure call.

Agent Greeting

Agent Greeting allows a CTI application (for example, the Contact Center) to instruct Unified Communications Manager to automatically play a pre-recorded announcement to the customer immediately after a successful media connection to the agent device. Applications are responsible for answering this call and playing greeting (for example, “Thank you for calling Citibank Visa. My name is Joe. May I have your account number please.”). The agent and customer can hear the agent greeting and the agent can remain on mute until the greeting ends or talk to the customer before the greeting ends. To support the Agent Greeting feature, Cisco Unified TSP introduces a new CCiscoLineDevSpecific extension to initiate and stop Agent Greeting.

To handle agent greetings during the customer calls:

- CCiscoLineDevSpecificStartSendMediaToBIBRequest allows the application to initiate an agent greeting to the customer call. The request contains IVRDN and CGPNToIVR.
- CCiscoLineDevSpecificStopSendMediaToBIBRequest allows the application to stop an agent greeting. This request is placed on the line where the agent greeting is currently playing and no other parameter is required.
- When an agent greeting is initiated or stopped, Cisco Unified TSP sends LineCallDevSpecific (SLDSMT_MEDIA_TO_BIB_STARTED) or LineCallDevSpecific (SLDSMT_MEDIA_TO_BIB_ENDED) to application.
- In the case of an agent greeting ended event, param2 indicates if the agent greeting was played successfully.
- If the application opens another line while an agent greeting is currently playing to the agent-to-customer call, Cisco Unified TSP exposes SendMediaToBIB bitmap in CallAttributeBits bitmap of

LineCallInfo::DevSpecific indicating that the agent greeting is in progress. When the agent greeting ends, Cisco Unified TSP sends LineCallDevSpecific (SLDSMT_MEDIA_TO_BIB_ENDED) to the application and clears SendMediaToBIB bitmap in the CallAttributeType field.

- When the agent greeting call arrives on the IVR port, two new bitmaps (ServerCall and BIBCall) are exposed as CallAttributeType. Cisco Unified TSP exposes LINECALLREASON_UNKNOWN as TAPI call reason when the agent greeting call arrives at IVR port, and exposes CtiReasonSendMediaToBIB in the ExtendedCallReason field.
- CTI Port and Route Point do not support Agent Greeting. Application gets LINEERR_OPERATIONUNAVAIL if CCiscoLineDevSpecificStartSendMediaToBIBRequest is issued on a CTI port or Route Point.

Note To support this feature, the application must negotiate line extension 0x000B0000 or above.

Interface Changes

See Start Send Media to BIB, Stop Send Media to BIB, Media to BIB Started Event, Media to BIB Ended Event, and Details.

Message Sequences

See Agent Greeting.

Backward Compatibility

This feature is backward compatible.

Agent Zip Tone

The Agent Zip Tone Support feature allows the TAPI applications to play tone on active calls, that is, play a tone on an agent phone after the call is answered (active state) by an agent.

TAPI defines a new line devspecific CCiscoLineDevSpecificPlaytone (lineHandle, callHandle, Tone, and PlayToneDirection) message type. Applications use this message type to play the tone on a phone placed locally or remotely.

Applications can play the tone on a local phone when the call is in the Accepted/Ringback state and on a remote phone when the call is in the Connected state.

When the application issues CCiscoLineDevSpecificPlaytone request with tone as CTONE_ZIP and direction as local, then the Zip tone is played at the local phone and the

LINE_DEVSPECIFIC event with the following parameters is reported on the local phone indicating that the tone is played:

- dwParam1 = SLDSMT_CALL_TONE_CHANGED
- dwParam2 = CTONE_ZIP
- dwParam3 = 0(local)

Similarly, when the application issues CCiscoLineDevSpecificPlaytone request with tone as CTONE_ZIP and direction as remote, then the Zip tone is played at the remote phone and the

LINE_DEVSPECIFIC event the following parameters is reported on the remote phone indicating that the tone is played:

- dwParam1 = SLDSMT_CALL_TONE_CHANGED
- dwParam2 = CTONE_ZIP
- dwParam2 = 0(local)

and also LINE_DEVSPECIFIC event the following parameters is reported on the local phone:

- dwParam1 = SLDSMT_CALL_TONE_CHANGED
- dwParam2 = CTONE_ZIP
- dwParam3 = 1(remote)

Extension 0x000B0000 is introduced for release 8.5(1).

Interface Changes

See Agent Zip Tone.

Message Sequences

See Agent Zip Tone.

Backward Compatibility

This feature is backward compatible.

Alternate Script

Certain IP phone types support an alternate language script other than the default script that corresponding to the phone configurable locale. For example, the Japanese phone locale associates two written scripts. Some phone types support only the default Katakana script, while other phones types support both the default Katakana script and the alternate Kanji script. Because applications can send text information to the phone for display purposes, they need to know what alternate script a phone supports – if any.

Arabic and Hebrew Language

Users can select Arabic and Hebrew languages during installation and also in the Cisco TSP settings user interface.

Barge and cBarge

Unified Communications Manager supports the Barge and cBarge features. The Barge feature uses the built-in conference bridge. The cBarge feature uses the shared conference resource.

Cisco Unified TSP supports the events that are caused by the invocation of the Barge and cBarge features. It does not support invoking either Barge or cBarge through an API of Cisco Unified TSP.

Call Control Discovery

The Call Control Discovery feature facilitates provisioning for inter-call agent communications. It uses the Service Advertisement Framework (SAF) network service to advertise itself as a call control entity and to discover other call control entities (CUCMs or CMEs) on the network so that it can dynamically adapt their routing behavior.

When call is made between two devices on different clusters, and the ICT bandwidth doesn't allow the call to go through, the CCD feature will fail over the call through a PSTN trunk to reach the same destination. PSTN failover will also be triggered by the CCDRequestingService when the call to a learned Hosted DNPattern gets rejected with a cause code other than unallocated, unassigned number and user busy

TAPI shall pass the CtiReasonSAF_CCD_PSTNFailover in the existing ExtendedCallReason field of the devSpecific portion of lineCallInfo. Since TAPI requires the TAPI call reason field to be set only once, the TAPI call reason shall remain as LINECALLREASON_DIRECT.

Interface Changes

There are no interface changes.

Message Sequences

See Call Control Discovery

Backward Compatibility

This feature is backward compatible.

Calling Party IP Address

The Calling Party IP Address feature provides the IP address of the calling party. The calling party device, which must be supported, must be an IP phone. The IP address is provided to applications in the devspecific data of LINECALLINFO. A value of zero (0) indicates that the information in not available.

The enhancement provides the IP address to the destination side of basic calls, consultation calls for transfer and conference, and basic redirect and forwarding. If the calling party changes, no support is provided.

Message Sequence

See Calling Party IP Address.

Calling Party Normalization

Prior to the Unified Communication Manager Release 7.0(1), the “+” symbol was not supported. Also, no support existed for displaying the localized or global number of the caller to the called party on its alerting display and the entry into its call directories for supporting a callback without the need of an EditDial.

Unified Communications Manager Release 7.0(1) adds support for "+" symbol and also the calling number is globalized and passed to the application. This enables the end user to dial back without using EditDial. Along with the globalized calling party, the user would also get the number type of the calling party. This would help the user to know where the call originated, that is, whether it is a SUBSCRIBER, NATIONAL or INTERNATIONAL number.

Interface Changes

See LINECALLINFO.

Message Sequences

See Calling Party Normalization.

Backward Compatibility

This feature is backward compatible.

Call PickUp

Call Pickup enables TAPI applications to invoke pickup, group-pickup, other-pickup, and directed pickup features from the application. Apart from providing the API to invoke Call PickUp feature, application registers Call pickup groups for alert notification, whenever a call is available for pickup. There will not be any notification if the call is picked up and the alerting stops.

Whenever there is a new call on the Pickup Group, TAPI fires a LINE_APPNEWCALL event followed by a LINE_CALLSTATE with a LINECALLSTATE_UNKNOWN | CLDSMT_CALL_PICKUP_STATE.

TAPI provides the pickup group Direct Number or Partition for the line in the devSpecific data of LINEDEVCAPS when LineGetDevCaps API is invoked with Extension version 0x000A0000 or higher.

New LineType is added for this feature, which is exposed to Application in Devspecific part of LINEDEVCAPS for the Pickup Line.

#define LINEDEVCAPSDEVSPECIFIC_PICKUPDN 0x00000004

New extension 0x000A0000 must be negotiated to use the new APIs.

Range of Permanent Line ID for the Pickup Line is between MAX_PICKUP_PERMID and MIN_PICKUP_PERMID.

const DWORD MAX_PICKUP_PERMID = 0xFFFFFFFF;

const DWORD MIN_PICKUP_PERMID = 0xFF000000;

New Call State Callpickup State is added for this feature

#define CLDSMT_CALL_PICKUP_STATE 0x10000000

Interface Changes

See RegisterCallPickUpGroupForNotification, UnRegisterCallPickUpGroupForNotification, and CallPickUpRequest.

Message Sequences

See Call PickUp

Backward Compatibility

This feature is backward compatible.

Call Queuing Feature Support

Unified Communications Manager queuing feature provides the ability to hold callers in a queue if there are more calls distributed through the call distribution feature than it can handle at any given time until the hunt members are available to answer them. While a call is in queue, the user is given the initial greeting announcement, music on hold, repeated announcements and so on.

TAPI exposes the below call reasons for the respective conditions in extendedCallInfo in the DevSpecific part of the LINECALLINFO structure.

- CallQueue {45 (0x2D)}

  CallQueue call reason is exposed on the calling party call when the call is queued when all HuntMembers of the HuntGroup are busy

- CallDeQueue {46 (0x2E)}

  CallDeQueue call reason is exposed on the connected party call when the dequeued-call is offered on the available Huntmember or any other DN which has been configured in the Queuing feature if no Huntmember is available

- CallDeQueueTimerExpired {47 (0x2F)}

  CallDeQueueTimerExpired call reason is exposed on the connected party call when the queued-call timer expires and the call is offered on the DN which has been configured in the Queuing feature when the "Maximum wait time in Queue" expires

- CallDeQueueAgentsBusy {48 (0x30)}

  CallDeQueueAgentsBusy call reason is exposed on the connected party call when all the hunt members are busy and hence the call is never queued and directly offered on the DN which has been configured in the Queuing feature when "Maximum Number of callers allowed in Queue" is reached

- CallDeQueueAgentsUnavailable {49 (0x31)}

  CallDeQueueAgentsUnavailable call reason is exposed on the connected party call when no hunt member is either logged-in or registered and the call is offered on the DN which has been configured in the Queuing feature when "No hunt members are Logged-in or registered"

TAPI shall not expose the ConnectedHuntPilotDN in the devspecific part of LINECALLINFO, on the calling party when the call is queued. The connectedHuntPilotDN is updated when the de-queued call offered on one of the hunt members is answered and goes to connected state.

The following Call queue setting configurations are available in the Hunt Pilot Configuration page.

Maximum Number of Callers Allowed in Queue (1-100): This is the queue depth configuration and reflects the maximum number calls that can be in the queue at any point of time.

Destination When Queue is full: It is the user configurable destination number to which the calls are forwarded when the maximum number of calls allowed in queue limit is reached.

Maximum Wait Time in Queue (10 -3600 seconds): User configurable maximum wait time a call on be in the queue.

Destination When Maximum Wait Time is Met: User configurable destination DN to which the call is forwarded when the maximum wait time in queue is reached.

Destination When There Are No Agents Logged In or Registered: User configurable destination DN to which the queue feature forwards the calls when none of the hunt members in the HuntPilot are registered or logged in.

Interface Changes

Not applicable.

Message Sequences

See Call Queuing.

Backward Compatibility

This feature is not backward compatible.

Call Recording and Call Recording Enhancement

The Call Recording feature provides two ways of recording the conversations between the agent and the customer: automatic call recording and selective call recording. A line appearance configuration determines which mode is enabled. Administrators can configure no recording, automatic recording of all calls, or selective recording for a line appearance. In selective call recording, recording can be initiated using a softkey or programmable line key assigned to the device, a CTI-enabled application, or both interchangeably.

Selective recording supports two modes: silent recording and user recording.

In the silent recording mode, the call recording status is not reflected on the Cisco IP device display. Silent recording is typically used in a call center environment to enable a supervisor to record an agent call. A CTI-enabled application running on the supervisor desktop is generally used to start and stop the recording for the agent-customer call.

In the user recording mode, the call recording status is reflected on the Cisco IP device display. A recording may be started or stopped using a softkey, programmable line key, or CTI-enabled application running on the user desktop.

The recording configuration on a line appearance cannot be overridden by an application. TSP will report ‘Recording type’ information to app in devSpecificData of LineDevCaps structure. Whenever there is a change in ‘Recording Type’, TSP will send LINE_DEVSPECIFIC (SLDSMT_LINE_PROPERTY_CHANGED with indication of LPCT_RECORDING_TYPE) event to application.

If the automatic call recording is enabled, a recording session will be triggered whenever a call is received or initiated from the line appearance. When the application invoked call recording is enabled, application can start a recording session by using CCiscoLineDevSpecificStartCallRecording (SLDST_START_CALL_RECORDING) on the call after it becomes active. The selective recording can occur in the middle of the call, whereas the automatic recording always starts at the beginning of the call. The

recorder is configured in CallManager as a SIP trunk device. Recorder DN can not be overridden by an application.

TSP will provide start recording request in lineDevSpecific to app for establishing a recording session. Application need to provide toneDirection as input to TSP in the start recording request. The result of the recording session is that the two media streams of the recorded call (agent-customer call) is being relayed from agent's phone to the recorder. TSP will provide agent's CCM Call Handle in the devSpecificData of LINECALLINFO.

TSP will inform the application when recording starts on its call by sending LINE_CALLDEVSPECIFIC (SLDSMT_RECORDING_STARTED) event. TSP will provide recording call attribute information (deviceName, DN, Partition) in devspecific data of LINECALLINFO after recording starts.

The recording session will be terminated when the call is ended or if app sends stop recording request to TSP through lineDevSpecific – CciscoLineDevSpecificStopCallRecording (SLDST_STOP_CALL_RECORDING).TSP will inform agent by sending LINE_CALLDEVSPECIFIC (SLDSMT_RECORDING_ENDED) when recording is stopped by stop recording request.

Both recording and monitoring get supported only for IP phones/CTI supported phones that are running SIP and within one cluster. It can be invoked only on phones that support built in bridges. Also built in bridge should be turned on to monitor or record calls on a device. Monitoring party does not need to have a BIB configured. Recording and monitoring will not be supported for secure calls in this phase.

Call Attributes

Call Attributes can be found in the DEVSPECIFIC portion of the LINECALLINFO structure. The Call Attribute Info is presented in the format of an array because Silent Monitoring and Call Recording could happen at the same time.

```
DWORD CallAttrtibuteInfoOffset;
DWORD CallAttrtibuteInfoSize;
DWORD CallAttrtibuteInfoElementCount;
DWORD CallAttrtibuteInfoElementFixedSize;
```

Offset pointing to array of the following structure:

```
typedef struct CallAttributeInfo{
    DWORD CallAttributeType;
    DWORD PartyDNOffset;
    DWORD PartyDNSize;
    DWORD PartyPartitionOffset;
    DWORD PartyPartitionSize;
    DWORD DeviceNameOffset;
    DWORD DeviceNameSize;
}CallAttributeInfo;
```

enum CallAttributeType

```
{
CallAttribute_Regular                    = 0,
CallAttribute_SilentMonitorCall,
CallAttribute_SilentMonitorCall_Target,
CallAttribute_RecordedCall,
CallAttribute_WhisperCoachingCall,
CallAttribute_WhisperCoachingCall_Target,
CallAttribute_Recorded_Automatic,
CallAttribute_Recorded_AppInitiatedSilent,
```

```
CallAttribute_Recorded_UserInitiatedFromApp,
CallAttribute_Recorded_UserInitiatedFromDevice
} ;
```

For recorded calls, if the application negotiates an extension less than 0x000C0000 the CallAttributeType is set to CallAttribute_RecordedCall. If the application negotiates an extension version equal to 0x000C0000 or higher, the CallAttributeType is set to CallAttribute_Recorded_Automatic, CallAttribute_Recorded_AppInitiatedSilent, CallAttribute_Recorded_UserInitiatedFromApp, or CallAttribute_Recorded_UserInitiatedFromDevice.

Call Recording Enhancement

Unified Communications Manager Release 9.0 the Call Recording feature is enhanced to allow user to start/stop current active call recording by pressing softkey on IP phone. Record key toggles between start and stop modes.

When TAPI Application invokes Recording Start / Stop APIs it has the same effect as if the user pressed the Record key on his IP phone. In addition, applications can control whether or not the phone display indication of on-going recording.

The data types that are used by Cisco TSP to report recording configuration and recording type to TAPI applications are re-worked in order to reflect the recent changes in UCM.

Interface Changes

- LineDevCaps::DevSpecific change (Cisco Extention 0x000C0000) LINEDEVCAPS
- LineCallInfo::DevSpecific change (Cisco Extention 0x000C0000) LINECALLINFO
- CciscoLineDevSpecificStartCallRecording Start Call Recording
- CciscoLineDevSpecificStopCallRecording StopCall Recording

Message Sequence

There are no changes to the message sequence with this enhancement.

Backward Compatability

This feature is backward compatible.

CallFwdAll Notification

This enhancement allows TAPI applications to distinguish off-hook calls (outgoing calls) from calls made by using the CFwdAll softkey.

TAPI provide two new additional mask bits in the existing bitmask, CallAttributeBitMask, as a part of LINECALLINFO::DEVSPECIFIC. For normal outgoing calls, the new mask is set to 0 and if a call is generated due to CFwdAll activation or deactivation, the corresponding new bit is set to 1.

Cases where the user presses CFwdAll softkey for an on-hook device are addressed, but when users go off-hook first and then press CFwdAll softkey, are not covered.

Interface Changes

In the CallAttributeBitMask field, LINECALLINFO::DEVSPECIFIC is modified to include the two new bit masks, TSPCallAttribute_CallForwardAllSet and TSPCallAttribute_CallForwardAllClear. For more information, see Details.

Message Sequences

See CallFwdAll Notification.

Backward Compatibility

This feature is backward compatible.

Cisco Unified TSP Auto Update

Cisco Unified TSP supports auto update functionality, so the latest plug-in can be downloaded and installed on a client machine. Be aware that the new plug-in will be QBE compatible with the connected CTIManager. When the Unified Communications Manager is upgraded to a newer version, and Cisco Unified TSP auto update functionality is enabled, the user will receive the latest compatible Cisco Unified TSP, which will work with the upgraded Unified Communications Manager. This ensures that the applications work as expected with the new release (provided the new Cisco Unified Communications Manager interface is backward compatible with the TAPI interface). The locally installed Cisco Unified TSP on the client machine allows applications to set the auto update options as part of the Cisco Unified TSP configuration. The user can opt for updating Cisco Unified TSP in the following different ways:

- Update Cisco Unified TSP whenever a different version (higher version than the existing version) is available on the Unified Communications Manager server.
- Update Cisco Unified TSP whenever a QBE protocol version mismatch exists between the existing Cisco Unified TSP and the Unified Communications Managerversion.

CIUS Session Persistency

Wireless devices introduced by Cisco such as CIUS have the capability to move between WiFi networks and also across WiFi and VPN networks (over 3G/4G) and still retain their registration with the same CiscoUCM. However, due to the change in the network the IP address of the device might undergo a change. The same scenario is applicable for docking and undocking of the CIUS phones.

To indicate this change in IP address of wireless devices such as Cius, TAPI will expose the changed IP address for lineDevices and phoneDevices through DEVCAPS structure.

TAPI will send notification to Applications on change in IP Address information on lineDevices of CIUS Device. LINE_DEVSPECIFIC notification is fired on all Open lines on a CIUS Device. TSP would fire LINE_DEVSPECIFIC event with param1 = SLDSMT_LINE_PROPERTY_CHANGED,param2 = LPCT_DEVICE_IPADDRESS.

TAPI will send notification to Applications on change in IP Address information on phoneDevices. PHONE_DEVSPECIFIC notification is fired on the phoneDevices. TSP would fire PHONE_DEVSPECIFIC event with param1 = CPDSMT_PHONE_PROPERTY_CHANGED_EVENT, param2 = PPCT_DEVICE_IPADDRESS.

TAPI will expose the changed IP address, IP Addressing Mode (IPv4, IPv6) of lineDevices in the devspecific data of LINEDEVCAPS when lineGetDevCaps API is invoked with Extension version 0x00090001 or higher.

TAPI will expose the changed IP address, IP Addressing Mode (IPv4, IPv6) of phoneDevices in the devspecific data of PHONEDEVCAPS when phoneGetDevCaps API is invoked withExtension version 0x00090001 or higher.

Note CIUS devices are SIP end points and like all other SIP end Points, they currently don't support IPV6.

Interface Changes

No interface changes.

Message Sequences

See CIUS Session Persistency.

Backward Compatibility

This feature is backward compatible.

Click to Conference

The Click to Conference feature enables users to create conferences from an Instant Messaging (IM) application without creating a consult call first. The Cisco TSP treats the feature as an existing conference model; however, when the conference is created or dropped, the CtiExtendedReason may come as Click2Conference.

Interface Changes

None.

Message Sequences

See Click to Conference.

Backward Compatibility

This feature is backward compatible.

CCMEncryption Enhancements

Starting with release Unified Communications Manager 10.0(1), Encryption method, which is used to encrypt the user login password, is enhanced. Older CiscoTSP clients (9.x or earlier) use Symmetric Key Encryption. Starting with rlease 10.x, the CiscoTSP client is enhanced to use a combination of Asymmetric and Symmetric Encryption mechanism. This enhancement provides more security for user credentials in non-secured connections.

Cisco recommends that applications/users upgrade Cisco TAPI clients to take advantage of this security enhancement.

To maintain backward compatibility from CTI, a new CTI Service Parameter is introduced - **Require Public key Encryption.**

The default value for this Service Parameter for this Release is **False**.

On False: CTI/CUCM allows applications/CiscoTSP clients using symmetric encryption method and Asymmetric Public key encryption method to connect with CUCM.

On True: Only CiscoTSP clients/applications which use asymmetric PublicKey Encryption Method will be able to open provider with Unified Communications Manager 10.x.

Cisco recommends that applications upgrade Cisco TAPI clients and set this service parameter to "true". The proposed plan for future releases is to set default value as true for this service parameter and later deprecate it to ensure that applications do not use older CiscoTSP Clients which use Symmetric Encryption method.

Interface Changes

There are no interface changes for this feature.

Message Sequence

CCMEncryption Enhancements

Backward Compatibility

As mentioned above, new CTI Service parameter is added to maintain backward compatibility.

Conference Enhancements

The Conference feature of Unified Communication Manager has been enhanced with the following functions:

- Allowing a noncontroller to add another party into an ad hoc conference.

  Applications can issue the lineGetCallStatus against a CONNECTED call of a noncontroller conference participant and check the dwCallFeatures before adding another party into the conference. The application should have the PREPAREADDCONF feature in the dwCallFeatures list if the participant is allowed to add another party.

- Allowing multiple conferences to be chained.

Be aware that these features are only available if the 'Advanced Ad-hoc Conference' service parameter is enabled on the Unified Communications Manager.

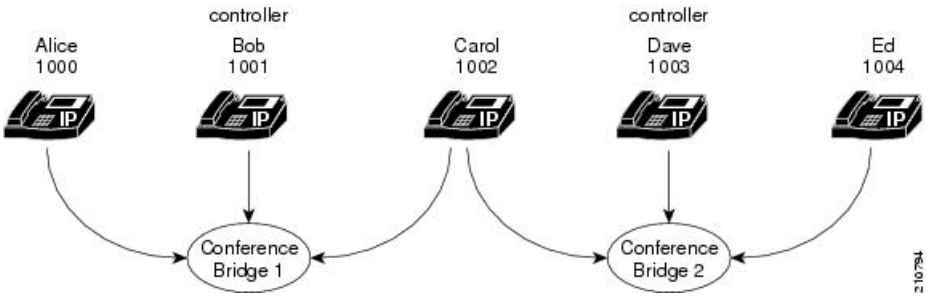
When this service parameter is changed from enabled to disabled, the system no longer allows new chaining between ad hoc conferences. However, existing chained conferences will stay intact. Any participant who is brought into the ad hoc conference by a noncontroller before this change will remain in the conference, but they can no longer add a new participant or remove an existing participant.

To avoid ad hoc conference resources remaining connected together after all real participants have left, Unified Communications Manager will disallow having more than two conference resources connected to the same ad hoc conference. However, using a star topology to connect multiple conferences could yield better voice quality than a linear topology. A new advanced service parameter, 'Non-linear Ad Hoc Conference Linking Enabled', lets an administrator select the star topology.

A participant can use the conference, transfer, or join commands to chain two conferences together. When two conferences are chained together, each participant only sees the participants from their own conference, and the chained conference appears as a participant with a unique conference bridge name. In other words, participants do not have a full view of the chained conference. The system treats the conferences as two separate conferences, even though all the participants are talking to each other.

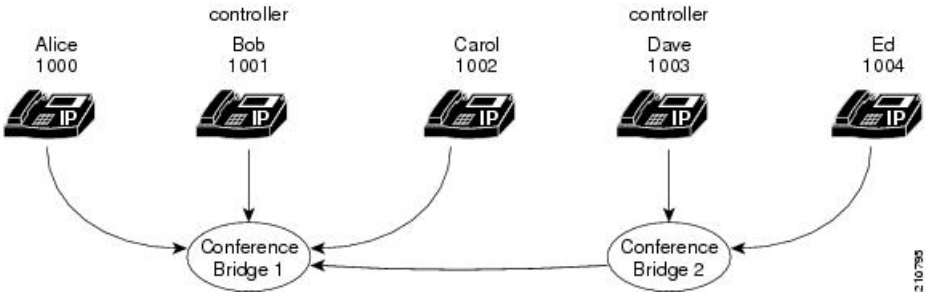
The following figures shows how TSP presents a conference model in the case of conference chaining. A, B, and C are in conference-1, and C, D, and E are in conference-2. C has an ONHOLD call on conference-1 and an active call on conference-2.

Figure 1: Conference Before Join



C then does a join with the primary call from conference-1. For A, B, and C, the conference participants comprise A, B, C, and conference-2. For D and E, the conference participants comprise D, E, and conference-1.

Figure 2: Conference After Join



When a user removes a CONFERENCE from its conference list on the phone, the operation actually drops the chained conference bridge. In the previous example, the two chained conferences have been unchained. Conference-1 will remain active and has A, B, and C as participants. However, conference-2 will become a direct call between Dave and Ed because they are the only two parties left in the conference.

Applications can achieve conference chaining by issuing a JOIN or TRANSFER on two separated conference calls. However, a LineCompleteTransfer with a conference option will fail due to a Microsoft TAPI limitation on this standard API. The application can use the Cisco LineDevSpecific extension to issue the join request to chain multiple conferences together.

Note As of Unified Communications Manager Release 8.6, Cisco TelePresence MCU conference bridges are supported through JTAPI/TSP. From a JTAPI/TSP perspective, these conference bridges behave in the same way as other supported conference bridges.

CTI Port Third-Party Monitoring Port

Opening a CTI port device in first-party mode means that either the application is terminating the media itself at the CTI port or that the application is using the Cisco Wave Drivers to terminate the media at the CTI port. This also comprises registering the CTI port device.

Opening a CTI port in third-party mode means that the application is interested in just opening the CTI port device, but it does not want to handle the media termination at the CTI port device. An example of this would be a case where an application would want to open a CTI port in third-party mode because it is interested in monitoring a CTI port device that has already been opened/registered by another application in first party mode. Opening a CTI Port in third-party mode does not prohibit the application from performing call control operations on the line or on the calls of that line.

Cisco Unified TSP allows TAPI applications to open a CTI port device in third-party mode via the lineDevSpecific API, if the application has negotiated at least extension version 6.0(1) and set the high order bit, so the extension version is set to at least 0x80050000.

The TAPI architecture lets two different TAPI applications that are running on the same PC use the same Cisco Unified TSP. In this situation, if both applications want to open the CTI port, problems could occur. Therefore, the first application to open the CTI port will control the mode in which the second application is allowed to open the CTI port. In other words, all applications that are running on the same PC, using the same Cisco Unified TSP, must open CTI ports in the same mode. If a second application tries to open the CTI port in a different mode, the lineDevSpecific() request fails.

CTI Remote Device

This feature provides the TSP/CTI applications to extend its ability to monitor and have limited call control capability over third-party devices of a User. This capability is provided to users by representing all the third-party devices/end points of a user as a Remote Destinations configured on a virtual device type named as "CTI Remote Device".

"CTI Remote Device" is a new type of Virtual Device, can be configured from Admin pages just like any other device and need to be associated with End User with Mobility support enabled. Remote Destinations can be configured on CTI Remote Device page and each of these Remote Destinations will be configured with the Number which can be used to dial any Third Party devices (PSTN, Mobile or other PBX) and thus each of these Remote Destinations will be representing one of the third-party devices of a User which are not actually registered to Unified Communications Manager.

"CTI Remote Device" can be configured with 5 Lines, each of them shared with Enterprise Phone's Lines. On any incoming call to CTI Remote Device, the call will be extended to all the Remote Destinations/Third Party Devices configured/associated with CTI Remote Device. Call will be reported to Applications on a line on CTI Remote Device, which represents the call that is extended to Third Party Devices. Call events will be reported like other normal calls representing the state of the calls extended to Third Party Devices.

"Cisco Unified Client Services Framework" (CSF) Devices are also enhanced to be registered in Extend Mode from Jabber Clients. In the Extend Mode CSF devices behave exactly like CTI Remote Devices. Capability to configure and associate Remote Destinations to CTI Remote Device is also extended to CSF Devices.

Remote Destinations can be configured from CUCM Admin Device pages of CTI Remote Device and CSF Device or from Remote Destination pages (add new and associate it to CTI Remote/CSF device) or from TSP applications using Add/Update/Remove Remote Destination feature. The max number of remote destinations

can be configured for a single CTI Remote Device depends on its owner user's max remote destinations configuration on the CUCM Admin user page (Default is 4).

Following are the supported features on CTI Remote Devices in this Release.

1. Receiving Incoming Enterprise Calls
2. Make Call (DVO -Dial via Office)
3. Call Disconnect
4. Hold / UnHold(Resume/Retrieve)
5. Redirect
6. DSS(Device State Server), and DND -(Do Not Disturb) and CPN (Globalized Calling Party)
7. Call Forwarding (Busy, Forward All ...)
8. Transfer and Direct Transfer
   a. only Direct Transfer on Same Line (DTSL) is supported
9. Conference and Join
   a. DropAnyParty Feature is also supported
   b. only Join on Same Line (JSL) is supported
10. Add/Update/Remove Remote Destinations
11. URI dialing

Refer to FFS for detailed information on feature (EDCS # 1048080).

To support this feature, CiscoTSP enumerates and exposes the newly added "CTI Remote Device" and expose the Remote Destination information configured on the "CTI Remote Device" to applications. New Line Type is added for CTI Remote Device, which is exposed to Application in Devspecific part of LINEDEVCAPS (LINEDEVCAPS::DevSpecific::dwLineTypeFlags = LINEDEVCAPSDEVSPECIFIC_REMOTEDEVICE (0x00000008)) using which Remote Device Lines are identified.

It provides applications/Users capability to Add new Remote Destinations and Remove/Update existing Remote Destinations configured on "CTI Remote Device" or CSF Devices. CiscoTSP would provide applications the capability to monitor and control incoming and outgoing calls.

Note Note: This feature requires a Cisco Jabber client and this functionality is intended to be supported in Jabber for Windows 9.1

Interface Changes

The following new interfaces were added to support this feature:

- CciscoLineDevSpecificAddRemoteDestination Add Remote Destination
- CciscoLineDevSpecificRemoveRemoteDestination Remove Remote Destination
- CciscoLineDevSpecificUpdateRemoteDestination Update Remote Destination

The following new error codes were added to support this feature:

Error	Description
LINEERR_DUPLICATE_INFORMATION	Reported on new LineDevSpecific Extensions (Add and Update) when the remote Destination Information which application is trying to update or add is already available.
LINEERR_REMOTE_DESTINATION_UNAVAIL	Reported on new LineDevSpecific Extensions (Update and Remove) when the RemoteDestination Number provided is not present on the Device/Line.
LINEERR_REMOTE_DESTINATION_LIMIT_EXCEEDED	Reported on new LineDevSpecifc Extension (Add) when total Remote Destination count has exceeded the limit of Remote Destinations configured for Owner User ID associated with Cti Remote Device/CSF device.
LINEERR_OPERATION_FAIL_NO_ACTIVE_RD_SET	Reported if any feature related operation request on CTI RemoteDevice failed as Active RD is not set.
LINEERR_ENDUSER_NOT_ASSOCIATED_WITH_DEVICE	Reported if any feature related operation request on CTI RemoteDevice failed as there is no Associated End User associated with Device.

See device specific extensions.

Message Sequences

See CTI Remote Device.

Backward Compatibility

This feature is backward compatible.

Application Dial Rule Support

Starting with Cisco Unified Communication Manager Release 9.1 a matching Application Dial Rule is applied prior to digit analysis when a call is offered to Remote Destination associated with a CTI Remote Device. When an application adding or updating Remote Destination on a CTI Remote Device is a part of verification process the Application Dial Rules are applied before digit analysis.

Interface Change

No Interface changes

Message Sequensce

Not Applicable

Backward Compatibility

In Cisco Unified Communication Manager Release 9.0 Application Dial Rules were not applied to Remote Destinations that are associated with CTI Remote Device.

DTMF Support

Starting with Cisco Unified Communication Manager Release 9.1, applications are able to invoke lineGenerateDigits() API on a CTI Remote Device. Only out-of-band DTMF is supported in Cisco Unified Communication Manager Release 9.1.

Interface Change

No Interface changes

Message Sequensce

Not Applicable

Backward Compatibility

This is a new feature and is backward compatible.

Extend Mode Support for CSF Is Removed

In Cisco Unified Communication Manager Release 9.0 a CSF device had the ability to add Remote Destinations, starting with Cisco Unified Communication Manager Release 9.1 CSF devices can no longer register with CTI in extend mode as CTI Remote Devices and CSF devices will not be able to add Remote Destination.

Interface Change

No Interface changes

Message Sequensce

Not Applicable

Backward Compatibility

Application upgrades from Cisco Unified Communication Manager Release 9.0 to Release 9.1 or later results in the removal of Remote Destinations configured for CSF devices.

Remote Destination Reachability Verification

In Cisco Unified Communication Manager Release 9.1, an enhancement is added in CTI to verify Remote Destination reachability when it is added or updated on a CTI Remote Device. To determine if the destination is reachable, CTI performs digit analysis based on Reroute CSS configured on the CTI Remote Device. The reachability verifies that the outside dial prefix, CSS and route pattern are configured correctly.

If the destination is not reachable the request returns the error: CTIERR_EXTEND_AND_CONNECT_DESTINATION_NOT_REACHABLE.

Error code

LINEERR_REMOTE_DESTINATION_NOT_REACHABLE – can be returned when an application attempts to add or update a Remote Destination and it cannot be reached.

Interface Change

No Interface changes

Message Sequensce

Not Applicable

Backward Compatibility

Backward compatibility issues may be seen when upgrading to Cisco Unified Communication Manager Release 9.1. In Unified Communication Manager Release 9.0 destination reachability was not verified and there were no errors returned.

Persistent Connection

Persistent connection or Persistent call refers to a call between a CTI remote device and a remote destination that stay connected even when no active customer calls exist. For example: At the beginning of the day, a Unified Communications Manager server phones a teleworker at home to establish a persistent connection that remains active all day, until the application drops the call.

At least one remote destination must be configured and active on the CTI remote device in order to create a persistent call. One persistent call for each remote device is allowed at a time. No feature invocations, such as park, hold, conference, and transfer, are allowed on persistent calls.

Once a persistent call is created it remains connected until application drops the call or the maximum call duration timer expires. A persistent call is also disconnected when a remote destination drops the call or is no longer active.

Persistent calls cannot be dropped if an active call to the remote device exists. Therefore, if there is an active call, the persistent call will be dropped as soon as that call is finished.

A Persistent Connection Call differs from a typical call in that no media events are generated for the persistent call. An application may not always receive notification about a persistent call that was accepted (LNECALLSTATE_ACCEPTED). This notification may depend on the type of trunks and gateways used in a specific telephone network.

The Persistent Call feature enhances some TAPI APIs and introduces new APIs and error codes.

Note This feature is backward compatible and existing applications are not affected.

Create a Persistent Call

The TAPI lineMakeCall function is used to create persistent call.. The relevant data is provided in LINECALLPARAMS structure pointed to by the lpLineCallParams parameter. Cisco TSP ignores all other lineMakeCall parameter for a persistent call.

For a persistent call, the LINECALLPARAMS contains the following data:

- DevSpecific part refering to Cisco_CallParamsDevSpecific structure where DevSpecificFlags is set to Cisco_CALLPARAMS_DEVSPECIFICFLAGS_PERSISTENT CALL (0x00000002)
- CallingPartyID set to a directory number that appears as a remote destination CallerID directory number

- DisplayableAddress set to a name that appears as the remote destination CallerIDName.

Drop a Persistent Call

Use the standard TAPI lineDrop function to drop or disconnect persistent call. The hCall parameter should specify the persistent call handle (HCALL) returned by lineMakeCall when the persistent call was created.

Persistent Call State Change

When the persistent call status changes, an application receives a standard TAPI LINE_CALLSTATE message. For a persistent call, the new call state in the dwParamas1 field will be constructed as follows:

- The low-order 24 bits are set to one of the LINECALLSTATE_constants as they are for a regular call.
- The high-order 8 bits is set to 0x20. A corresponding bitmask is defined in CiscoLineDevspecificMsg header file as CLDSMET_PERSISTENT_CALL_STATE (0x200000000).

Persistent Call Attribute Bit Mask

A new bit definition, TSPCallAttribute_PersistentCall(0x00004000), is added to the CallAttributeBitMask enumeration in CiscoLineDevSpecificMsg.h header file. For a persistent call, a corresponding bit is turned on in the CallAttributeBitMask field in Cisco TSP extension of the TAPI LINECALLINFO structure.

Interface Changes

- lineMakeCall TAPI Line Functions – lineMakeCall – Note added
- LINECALLINFO DevSpecific change - (Cisco Extention 000D0000 – new bit definition added for Call Attribute Type - TSPCallAttribute_PersistentCall(0x00004000))
- LINECALLPARAMS Cisco Device-Specific Extentions – LINECALLPARAMS section added.

The following new error codes were added to support this feature:

Error	Description
LINEERR_PERSISTENT_CALL_CREATE_FAILED	Attempt to create persistent call failed.
LINEERR_PERSISTENT_CALL_ALREADY_EXISTS	Persistent call cannot be created because another one already exists.
LINEERR_OPERATION_NOT_ALLOWED_ON_PERSISTENT_CALL	No feature invocation is allowed on persistent call, such as park, hold, conference, and transfer.
LINEERR_PERSISTENT_CALL_DROP_FAILED_CALL_ACTIVE	An attempt to disconnect persistent call while customer call is still active
LINEERR_NO_PERSISTENT_CALL_EXISTS	An attempt to create announcement call on a device where persistent call does not exist
LINEERR_PERSISTENT_CALL_NOT_ESTABLISHED	An attempt to create announcement call while persistent call is still being setup
LINEERR_OPERATION_NOT_AVAILABLE_IN_CURRENT_STATE	Requested operation is not allowed in current call state

Usage Cases

See Persistent Connection Use Cases.

Backward Compatibility

This enhancement is backward compatible and existing applications will not be affected with introduction of this enhancement.

Announcement Call

Cisco Extend and Connect is enhanced with an ability to play announcements to a remote destination. In order to play the announcement, an application creates a special type of call -Announcement Call -and identifies an announcement to be played.

The announcement can be played only to a remote destination with an existing Persistent Call. Only announcements that were uploaded to the Unified Communications Manager can be played.

Applications are notified that an announcement started by the LINE_DEVSPECIFIC event: SLDSMT_ANNOUNCEMENT_STARTED.

Applications are notified that an announcement stopped by the LINE_DEVSPECIFIC event: SLDSMT_ANNOUNCEMENT_ENDED.

Create Announcement Call

The TAPI lineMakeCall function is used to create announcement call. The relevant data is provided in LINECALLPARAMS structure pointed to by the lpLineCallParams parameter. CiscoTSP ignores all other lineMakeCall parameters in the case of announcement call.

In the case of an announcement call, the following data is provided in the LINECALLPARAMS:

- DevSpecific part refers to Cisco_CallParamsDevSpecific structure where the DevSpecificFlags is set to Cisco_CALLPARAMS_DEVSPECIFICFLAGS_ANNOUNCEMENTCALL
- CallData is set to a media content identifier (announcementID)

Drop Announcement Call

The standard TAPI lineDrop function drops or disconnects the announcement call. The hCall parameter specifies the announcement call handle (HCALL) returned by lineMakeCall when the announcement call is created.

Announcement Call State Change

When the status of the announcement call changes, an application receives the standard TAPI LINE_CALLSTATE message. For an announcement call, the construction of the new call state in the dwParams1 field is:

- The low-order 24 bits are set to one of the LINECALLSTATE_constants, which is the same as a regular call.
- The high-order 8 bits are set to 0x40. A corresponding bitmask is defined in the CiscoLineDevspecificMsg header file as follows:

  CLDSMT_ANNOUNCEMENT_CALL_STATE 0x40000000

Announcement Call Attribute Bit Mask

A new bit definition, TSPCallAttribute_AnnouncementCall (0x00008000), is added to the CallAttributeBitMask enumeration in the CiscoLineDevSpecificMsg.h header file. For an announcement call, a corresponding bit is turned on in the CallAttributeBitMask field in the Cisco TSP extension of the TAPI LINECALLINFO structure.

Interface Changes

- **Cisco_LineCallInfo_Ext000D0000extD0** is added. See LINECALLINFO.
- Cisco_CALLPARAMS_DEVSPECIFICFLAGS_ANNOUNCEMENTCALL is added. See LINECALLPARAMS.
- lineMakeCall:

  Beginning with Unified Communications Manager Release 10.01, applications can use lineMakeCall to create a Persistent Call or Announcement Call. For a Persistent Call or Announcement Call, the relevant data is provided in the LINECALLPARAMS structure pointed to by the lpLineCallParams parameter. All other lineMakeCall parameters are ignored in these cases.

- See Announcement Events.

Message Sequence

See Announcement Call.

Backward Compatibility

This enhancement is backward compatible and existing applications are not affected by the introduction of this enhancement.

NuRD (Number Matching for Remote Destination) Support

In CUCM 10.0, the existing "Cisco Extend and Connect" feature includes number matching for remote destination support. When users directly call a number that is configured as a remote destination for CTI Remote Device (CTI RD), and that remote destination is set to be active, the call is offered on the CTI Remote Device and extended to the remote destination. The called party is presented to the application as the CTI RD. If active remote destination is not set, when users call a remote destination number, a direct call between the caller and the remote destination occurs. This scenario also applies to a remote destination making a call to an enterprise directory number. If the remote destination is set to be active, from an application perspective, the CTI RD appears to initiate the call to the enterprise dn. If the active remote destination is not set, when the remote destination calls an enterprise dn, it is a direct call between the remote destination and the enterprise dn.

For those calls from and to a remote destination number, all existing features allowed on CTI RD can be performed.

Interface Changes

There are no interface changes for this feature.

Use Cases

See NuRD (Number Matching for Remote Destination) Support.

Backward Compatibility

This feature can change the existing expected behavior in regards to calls to and from remote destination numbers directly. Applications that do not want to leverage this NuRD feature can keep the cluster-wide service parameter "Reroute Remote Destination Calls to Enterprise Number" set to false. Enabling it will enable the NuRD features. This parameter by default is set to false.

Mobility Interaction Support

The "Cisco Extend and Connect" feature includes mobility interaction. Users can specify remote destinations that are shared between the CTI Remote Device (CTI RD) and the Remote Destination Profile (RDP). When both the CTI RD and the RDP are configured for the same user, and if the application is active (active rd is set), the CTI RD processes the call first and then offers the call to the RDP. If the application is not active, the RDP processes the call first and does not offer the call to the CTI RD. When only the CTI RD is configured for a user, the existing "Cisco Extend and Connect" feature behavior with remote destinations remains unchanged. When only RDP is configured for a user, there is no application support because the devices are not controllable from a CTI.

Interface Changes

There are no interface changes for this feature.

Use Cases

There are no new use cases for this feature. (Add/Update/Delete Remote Destination) are added for CTIRD, and are applicable (without any change from App/User).

Backward Compatibility

There are no backward compatibility issues for this feature.

Call Forwarding

Starting with Unified Communications Manager 10.0(1), a new feature called “CTI Rd Call Forward” allows users to control when incoming calls are forwarded to all configured Remote Destinations on the CTI Remote Device, when no active remote destination is set.

A new check box, **Route calls to all remote destinations when client is not connected**, is added to the Unified Communications Manager device window. The check box determines whether calls are routed to all remote destinations when Active Remote Destination is not set.

When you enable the **Route calls to all remote destinations when client is not connected** check box, and Active Remote Destination is not set, the call is routed to all remote destinations. If this check box is disabled, and Active Remote Destination is not set, the call is disconnected with User_Busy error on the CTI Remote Device.

In scenarios where Active Remote Destination is set, the call is always routed to the Active Remote Destination regardless of whether the **Route calls to all remote destinations when client is not connected** check box is enabled or disabled.

Interface Changes

There are no interface changes for this feature.

Message Sequence

See CTI RD Call Forwarding.

Backward Compatibility

There are no backward-compatibility issues for this feature.

CTI Video Support

The CTI Video Support feature allows the TAPI Application to retrieve the multimedia capabilities of Line Devices. Applications that monitor devices use this information to answer or route video calls to video capable devices.

TAPI shall expose a new structure DeviceMultiMediaCapability in Devspecific data of linedevcaps when applications issue a TSPI_LineGetDevCaps () API with Extension version 0x000D0000 or higher, on these line devices, Cisco TSP will fire **SLDSMT_LINE_PROPERTY_CHANGED** or **CPDSMT_PHONE_PROPERTY_CHANGED_EVENT** with param1= **LPCT_DEVICE_MULTIMEDIACAP_INFO** or **PPCT_DEVICE_MULTIMEDIACAP_INFO** to report Multimedia capability information change.

A new structure, DeviceCallMultiMediaCapInfo, is introduced under CiscoLineDevSpecificMsg.h, which provides the information about calling and called party multimedia capabilities that is exposed in the devspecific data of LineGetCallInfo.

Similarly, when the video capability of a calling/called of a Call changes, TSP fires **LINE_CALLDEVSPECIFIC** with param1= **SLDSMT_LINECALLINFO_DEVSPECIFICDATA** and param2= **SLDST_DEVICE_VIDEO_CAP_INFO** to report Multimedia capability information of the call

Also, devspecific data of LineCallInfo contains two new fieldsCallingPartyMultiMediaCapBitMask and CalledPartyMultiMediaCapBitMask, which indicate the fields in the DeviceMultiMediaCapInfo with valid information.

When the application makes a video call from one video enabled phone to another, the TSP fires **LINE_CALLDEVSPECIFIC** event with param1= **SLDSMT_MULTIMEDIA_STREAMSDATA** to report MultiMedia Streams information of the call. The Multimedia Streams information of the call is exposed in the Devspecific data of linecallinfo (As a part of VideoStreamInfo structure) when application issues TSPI_LineGetCallInfo() API with Extension version 0x000D0000 or higher.

The following table describes the video capabilities provided by TAPI for currently supported devices.

Device	Supports Initial Device Multimedia Capability	Supports Dynamic Video Capability Change	Reports calling and called Multimedia Capabilities on call info (LineGetCallInfo)	Supports Multimedia Streams Information
8945 (SIP)	Yes	Yes	Yes	Yes
8945 (SCCP)	Yes	Yes	Yes	No

Device	Supports Initial Device Multimedia Capability	Supports Dynamic Video Capability Change	Reports calling and called Multimedia Capabilities on call info (LineGetCallInfo)	Supports Multimedia Streams Information
9951, 9971(SIP)	Yes	Yes	Yes	Yes
EX60/90 (SIP)	Yes	N/A	Yes	Yes
CTS 500-32 (SIP)	Yes	N/A	Yes	Yes
Jabber(CSF/softphone mode) (SIP)	Yes	Yes	Yes	No
CTI RoutePoint (SCCP)	N/A	N/A	Yes	No
CTI Port (SCCP)	N/A	N/A	Yes	No
All other phones	N/A	N/A	Yes	No

Supported Features (With in the same cluster):

- Originating Call and Consult Call
- Redirect
- Call Forward
- Hold and Resume
- Hunt List
- Transfer
- Extension Mobility
- Super Provider

Supported Features (Across the cluster):

- Originating Call and Consult Call
- Redirect
- Call Forward
- Hold and Resume
- Hunt List
- Extension Mobility
- Super Provider

Limitations:

- Remote In Use
    - CiscoTSP does not provide correct calling and called party multimedia capabilities on a call that is in inactive state or is in Remote In Use state.
- MultiMedia Capability
    - Calling and called party multimedia capabilities are UKNOWN on the calling side until the called party answers the call.
    - When a call is initiated over SIP trunk configured with early offer, the called party video capabilities are the negotiated capabilities that get reported instead of the actual capability, on the called party.

- Only video capability information is known for calls over the H323 trunk, Screen count and telepresence interop information is unknown.

- MultiMediaStreams
  - CiscoTSP does not provide multimedia streams information if the device is a SCCP phone. The CiscoTSP does not deliver SLDSMT_MULTIMEDIA_STREAMSDATA, and the TSPI_LineGetCallInfo() API does not provide multimedia streams information in the VideoStreamInfo structure.
- Change in called party
  - In scenarios such as Shared Lines or redirect, where the called party changes, the application is notified of the new called party capability only if the called party is configured with unique display names.

Interface Changes

- LineDevCaps::DevSpecific change - (Cisco Extention 000D0000) LINEDEVCAPS
- LineCallInfo::DevSpecific change - (Cisco Extention 000D0000) LINECALLINFO
-
- LPCT_DEVICE_MULTIMEDIACAP_INFO – Indicates or notifies application that Device Multi Media Capability Information on the Line/Device has changed. Line Property Changed Events
- PPCT_DEVICE_MULTIMEDIACAP_INFO – Indicates or notifies application that Device Multi Media Capability Information on the Line/Device has changed. Phone Property Changed Events
- SLDST_DEVICE_VIDEO_CAP_INFO – (New bit mask type added for Param2 bits on SLDSMT_LINECALLINFO_DEVSPECIFICDATA) LINECALLINFO_DEVSPECIFICDATA Events
- SLDSMT_MULTIMEDIA_STREAMSDATA – (New Message Type in Line_DevSpecific Message) MultiMedia Streams Data Notification Event

Message Sequences

See Video Capabilities and Multimedia Information.

Backward Compatibility

This feature is not backward compatible.

Default CTI IP Addressing for Devices

A new CTIManager service parameter, **IP Addressing Mode for Devices**, has been added that allows you to configure the default CTI IP addressing mode for a device that does not have an associated Common Device Configuration.

Cisco TAPI communicates the value of this parameter to an application in the device-specific extension of the TAPI LINEDEVCAPS structure. When the application invokes the lineGetDevCaps() method for a device that does not have a Common Device Configuration, Cisco TAPI returns the value in dwLineDevCapsIPAddressingMode field. By default this parameter is set to allow both IPv4 and IPv6 modes (IPAddress_IPv4_IPv6).

Note For an individual CTI device, if that device has an associated Common Device Configuration, the IP Addressing Mode setting in the Common Device Configuration overrides the value of the **IP Addressing Mode for Devices** service parameter.

Device State Server

The Device State Server feature provides accumulative state of all the lines on the device. Applications are notified about the device status through the PHONE_DEVSPECIFIC and LINE_DEVSPECIFIC events.

An application for enabling the Device State Server support needs to set the DEVSPECIFIC_DEVICE_STATE and DEVSPECIFIC_DEVICE_STATE_STATUS_ message flags using the lineDevSpecific SLDST_SET_STATUS_MESSAGES request and the PhoneDevSpecific CPDST_SET_DEVICE_STATUS_MESSAGES request respectively.

When Cisco TSP receives the DEVICE_STATE events from CTI, it notifies the application about the accumulative state of all the lines on the device using the PHONE_DEVSPECIFIC and LINE_DEVSPECIFIC events.

The device status in the LINE_DEVSPECIFIC and PHONE_DEVSPECIFIC events can be one of the following.

```
enum lineDeviceState{
lineDeviceState_UNKNOWN = 0,
lineDeviceState_ACTIVE = 1,
lineDeviceState_ALERTING = 2,
lineDeviceState_HELD = 3,
lineDeviceState_WHISPER = 4,
lineDeviceState_IDLE = 5
};

enum PhoneDeviceState{
     PhoneDeviceState_UNKNOWN  = 0,
     PhoneDeviceState_ACTIVE = 1,
     PhoneDeviceState_ALERTING = 2,
     PhoneDeviceState_HELD = 3,
     PhoneDeviceState_WHISPER = 4,
     PhoneDeviceState_IDLE = 5
};
```

This feature provides the accumulative state of all the lines on the device or the phone to the application. Events are provided based on the following criteria:

- IDLE

  If all the lines on the device are IDLE, the device state is considered IDLE and the corresponding event is delivered to the qualified applications.

- ACTIVE

  If any of the lines on the device have an ACTIVE (call states are LINECALLSTATE_DIALTONE, LINECALLSTATE_DIALING, LINECALLSTATE_PROCEEDING, LINECALLSTATE_RINGBACK, and LINECALLSTATE_DISCONNECTED) call, the device state is considered ACTIVE and the corresponding event is delivered to the qualified applications.

- ALERTING

If there is no ACTIVE call on any of the lines of the device and at least one of the lines has an ALERTING (call states are LINECALLSTATE_OFFERING and LINECALLSTATE_ACCEPTED) call, the device state is considered ALERTING and the corresponding event is delivered to the qualified applications.

- HELD

  If there is no ACTIVE or ALERTING call on any of the lines of the device and at least one of the lines has a HELD call, the device state is considered HELD and the corresponding event is delivered to the qualified applications.

- WHISPER

  If there is no ACTIVE or ALERTING or HELD call on any of the lines of the device and at least one of the lines have an intercom call, the device state is considered WHISPER and the corresponding event is delivered to the qualified applications.

Note The ACTIVE state has priority over the ALERTING, HELD, and IDLE states.

The ALERTING state has priority over the HELD and IDLE states.

The HELD state has priority over the IDLE state.

Note To make the LineDevSpecific event indicate the device state for any line of that device, the DEVSPECIFIC_DEVICE_STATE_STATUS_message flag for that line must be turned on using the lineDevSpecific SLDST_SET_STATUS_MESSAGES request.

Direct Transfer

In Unified Communications Manager, the Direct Transfer softkey lets users transfer the other end of one established call to the other end of another established call, while dropping the feature initiator from those two calls. Here, an established call refers to a call that is either in the on hold state or in the connected state. The “Direct Transfer” feature does not initiate a consultation call and does not put the active call on hold.

A TAPI application can invoke the “Direct Transfer” feature by using the TAPI lineCompleteTransfer() function on two calls that are already in the established state. This also means that the two calls do not have to be set up initially by using the lineSetupTransfer() function.

Direct Transfer Across Lines

The Direct Transfer Across Lines feature allows the application to directly transfer calls across the lines that are configured on the device. The application monitors both the lines when directly transferring the calls across the lines.

A new LineDevSpecific extension, CciscoLineDevSpecificDirectTransfer, is added to direct transfer calls across the lines or on the same line. The 0x00090000 extension must be negotiated to use CciscoLineDevSpecificDirectTransfer.

Interface Changes

See Direct Transfer.

Message Sequences

See CTI Remote Device.

Backward Compatibility

This feature is backward compatible.

Directory Change Notification

The Cisco Unified TSP sends notification events when a device has been added to or removed from the user-controlled device list in the directory. Cisco Unified TSP sends events when the user is deleted from Unified Communications Manager.

Cisco Unified TSP sends a LINE_CREATE or PHONE_CREATE message when a device is added to a users control list.

It sends a LINE_REMOVE or PHONE_REMOVE message when a device is removed from the user controlled list or the device is removed from database.

When the system administrator deletes the current user, Cisco Unified TSP generates a LINE_CLOSE and PHONE_CLOSE message for each open line and open phone. After it does this, it sends a LINE_REMOVE and PHONE_REMOVE message for all lines and phones.

Note Cisco Unified TSP generates PHONE_REMOVE / PHONE_CREATE messages only if the application called the phoneInitialize function earlier.

The system generates a change notification if the device is added to or removed from the user by usingUnified Communications Manager or the Bulk Administration Tool (BAT).

If you program against the LDAP directory, change notification does not generate.

Do Not Disturb

The Do Not Disturb (DND) feature lets phone users go into a Do Not Disturb state on the phone when they are away from their phone or simply do not want to answer incoming calls. The phone softkey DND enables and disables this feature.

From the Unified Communications Manager user windows, users can select the DND option DNR (Do Not Ring).

Cisco TSP makes the following phone device settings available for DND functionality:

- DND Option: None/Ringer off
- DND Incoming Call Alert: Beep only/flash only/disable

- DND Timer: a value between 0-120 minutes. It specifies a period in minutes to remind the user that DND is active.
- DND enable and disable

Cisco TSP includes DND feature support for TAPI applications that negotiate at least extension version 8.0 (0x00080000).

Applications can only enable or disable the DND feature on a device. Cisco TSP allows TAPI applications to enable or disable the DND feature via the lineDevSpecificFeature API.

Cisco TSP notifies applications via the LINE_DEVSPECIFICFEATURE message about changes in the DND configuration or status. To receive change notifications, an application must enable the DEVSPECIFIC_DONOTDISTURB_CHANGED message flag with a lineDevSpecific SLDST_SET_STATUS_MESSAGES request.

This feature applies to phones and CTI ports. It does not apply to route points.

Do Not Disturb-Reject

Do Not Disturb (DND) enhancements support the rejection of a call. The enhancement Do Not Disturb–Reject (DND–R) enables the user to reject any calls when necessary. Prior to the Unified Communications Manager Release 7.0(1), DND was available only with the Ringer Off option. If DND was set, the call would still get presented but without ringing the phone.

To enable DND–R, access the Unified Communications Manager Administration phone page or the user can enable it on the phone.

However, if the call has an emergency priority set, the incoming call is presented on the phone even if the DND–R option is selected. This will make sure that emergency calls are not missed.

Feature priority is introduced and defined in the enum type for making calls or redirecting existing calls. The priority is defined as:

```
enum CiscoDoNotDisturbFeaturePriority {
CallPriority_NORMAL = 1
CallPriority_URGENT = 2
CallPriority_EMERGENCY = 3
};
```

Feature priority introduces LineMakeCall as part of DevSpecific data. Currently the following structure is supported in DevSpecific data for LineMakeCall:

```
typedef struct LineParams {
 DWORD FeaturePriority;
} LINE_PARAMS;
```

The new Cisco LineDevSpecific extension, CciscoLineRedirectWithFeaturePriority with type SLDST_REDIRECT_WITH_FEATURE_PRIORITY, supports redirected calls with feature priority.

Also in a shared line scenario, if one of the lines is DND–R enabled and if the Remote In Use is true, then it will be marked as connected inactive.

Interface Changes

See lineMakeCall and Redirect with Feature Priority.

Message Sequences

See Do Not Disturb-Reject.

Backward Compatibility

This feature is backward compatible.

Drop-Any-Party

The Drop-Any-Party feature enables the application to drop any call from the ad-hoc conference. This feature is currently supported from the phone interface. The application uses the LineRemoveFromConference function to drop the call from a conference. When the call is dropped from a conference, TSP receives CtiDropConferee as the call state change cause, and this is sent to TAPI as the default cause.

Interface Changes

See lineRemoveFromConference.

Message Sequences

See Drop Any Party.

Backward Compatibility

This feature is backward compatible. The 0x00090000 extension is added to maintain backward compatibility.

Early Offer

The Early Offer feature allows the SIP trunk to support early offer outbound calls without using MTP when the media capabilities and media port information of the calling endpoint is available. For the endpoints where the media port information is not available (for example, H323 slow start calls or delayed offer SIP calls or legacy SCCP phones) for Early Offer, Unified Communications Manager allocates an MTP to provide an offer. This means Unified CM allocates MTP only when needed.

To support the Early Offer feature, Cisco TSP introduces CCiscoLineDevSpecific extension (CciscoLineDevSpecificEnableFeatureSupport) to allow the application to enable or disable the Early Offer feature. This DevSpecific type is generic and can be used for supporting features added in the future.

The registration of CTI ports or route points are as follows:

- Dynamic registration of CTI ports or route points with Early Offer Support:
  - New LineDevSpecific type must be requested before registration.
- Static registration of CTI ports with Early Offer Support:
  - New LineDevSpecific type is requested before registration and used to change the Device Capability of Early Offer Support after registration.

GET IP and PORT EVENT reports to a CTI Port or Route Point registered with Early Offer Support enabled, on an outbound call. When an outbound call is routed through the SIP trunk with Early Offer Support, TSP reports LINE_DEVSPECIFIC event with Param1 = SLDSMT_RTP_GET_IP_PORT and Param2 = IPAddressing Mode along with SetRTP bit information (ninth bit from LSB).

dwParam2 = 0x00000xyy, where:

- x (ninth Bit from LSB) — SetRTPInfo (1 — Applications must set the RTP information and 0 — Applications must not set the RTP Information
- yy (8 bits) — IPAddressing Mode.

For dynamically-registered CTI ports or route points with Early Offer: For this notification, applications have to set the RTP information using the Existing LineDevspecific Type (CciscoLineDevSpecificSetRTPParamsForCall) with the IP and Port Information for the IPAddressing Mode reported. Applications must not set the RTP information on the Open Logical Channel notification if the application has already set the information on GetIP and Port notification (SLDSMT_RTP_GET_IP_PORT).

For statically-registered CTI ports: For this notification, applications must open and reserve the port used for registration.

To receive the Get IP and Port notification (SLDSMT_RTP_GET_IP_PORT), an application must set the DEVSPECIFIC_GET_IP_PORT message flag by using the lineDevSpecific SLDST_SET_STATUS_MESSAGES request.

TAPI provides setRTP information in dwParam2 of OpenLogical Channel notification (LINE_DEVSPECIFIC Event with dwParam1 = SLDSMT_OPEN_LOGICAL_CHANNEL) along with the IP addressing capability using which the application must determine whether it has to set the RTP information.

dwParam2 = 0x0000xxyy, where:

- xx — SetRTPInfo (1 — Applications must set the RTP information and 0 — Applications must not set RTP information)
- yy — IPAddressing Capability.

TSP Reports New Error Code (LINEERR_REGISTER_GETPORT_SUPPORT_MISMATCH) when application tries to dynamically or statically register CTI port or route point without Early Offer Support, where as the CTI port is already Registered Dynamically/Statically with Early Offer support by other applications.

Media Driver Support for Early Offer

For an Early Offer call on a CTI Port registered with Early Offer support, when the other party has the IP and Port information of the calling party, the other party starts transmitting the media early even before the Media Events are reported on the CTI Port (registered with Early Offer).

Due to this Early Media or delay in reporting Media Reception Event to the application, Wave Driver misses initial data transmitted as the current Wave Drivers (both Legacy and Cisco New Wave Driver) supports opening of the Ports and starts reception of data only after Media Events are reported to the application.

To capture the early transmitted data, the receiving port needs to be opened after GET_IP_PORT Event and Buffer Incoming Data. Current supported APIs for opening a port in New Wave Driver and Legacy Wave Driver requires CODEC and other supported information (DSCP, SRTP Information, and Silence Type). But in the Early Offer case, this information is not available at GET_IP_PORT event and needs to add new API's to open the Port, start buffering, and then Update the endpoint with other media endpoint related information (CODEC, DSCP, SRTP Information, and Silence Type).

The following API's have been added to the New Wave Driver to capture Early Media for Early Offer Call:

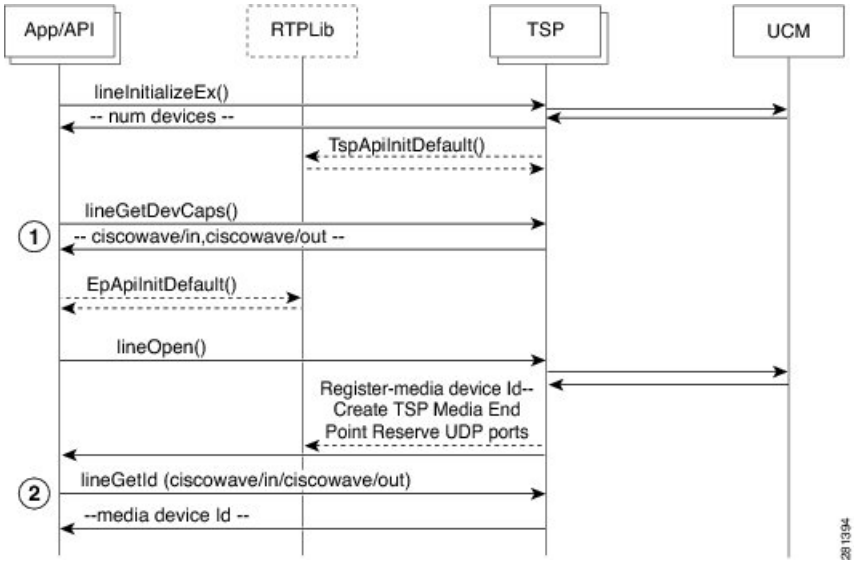
- EpStreamOpen()—Opens the Port and Starts Buffering Incoming Data.
- EpUpdateById()—Updates the Media endpoint Data Information (CODEC, DSCP, SRTP Information, and Silence Type).
  - Returns True on successful updation or false on failure; specific error code can be retrieved by calling EpApiGetLastError.
  - When the Stream is already started using EpStreamStart() API, EpUpdateById() request fails with error EP_ERR_TOAPP_INVALID_STATE.
  - When the Port is opened using EpStreamOpen() API, EpUpdateByID() will update the data information related to the media endpoint except the address and port.
  - On a Stream that is opened; In case of mismatch of LocalAddrInfo with the actual port used for Opening Socket, the request fails with error EP_ERR_ADDR_MISMATCH.

Note Applications must use these newly added APIs to capture Early Media Data for Early Offer Call.

TAPI Application Message Flow for Early Offer Call

The message flow in the following figure is described in steps 1 and 2.

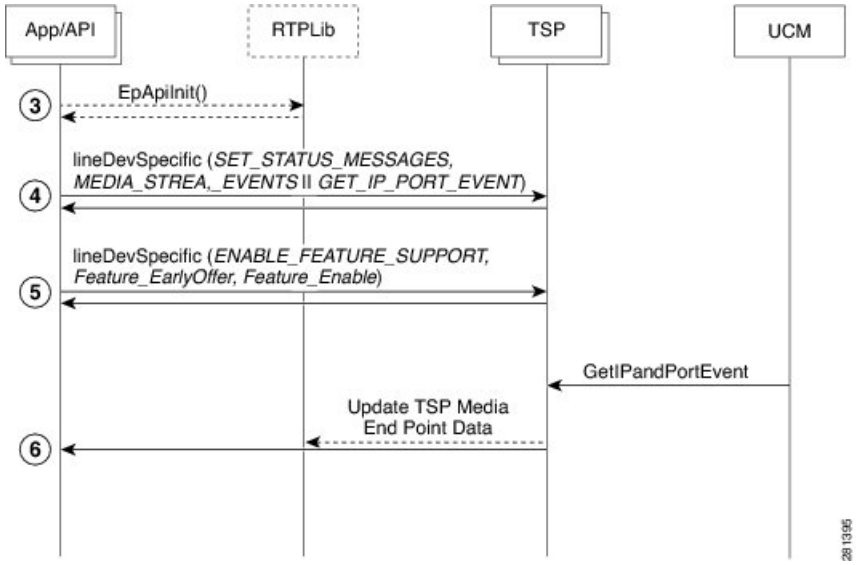
Figure 4: Application Message Flow for Early Offer Call — Steps 1 and 2



1. Initialize TAPI, get LINEINFO for the available line devices, and find the devices that are capable of using the Cisco RTP Library functionalities.
2. Get the media device identifier associated with a particular line device.

   The message flow in the following figure is described in steps 3 to 6.

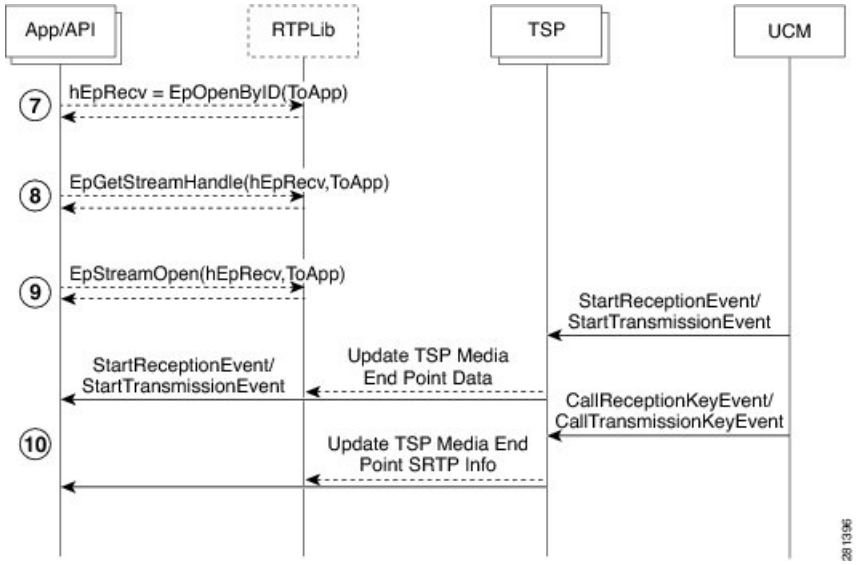
Figure 5: Application Message Flow for Early Offer Call — Steps 3 to 6



3. Initialize RTP Library.
4. Subscribe for media stream, and GetIP and Port events for the relevant devices using the Cisco lineDevSpecific extension (CciscoLineDevSpecificSetStatusMsgs).
5. Enable the Early Offer feature support on that line/device using the lineDevSpecific extension (CciscoLineDevSpecificEnableFeatureSupport).
6. GetIP and Port events reported to the application, and reports for the Early Offer call.

   The message flow in the following figure is described in steps 7 to 8.

Figure 6: Application Message Flow for Early Offer Call — Steps 7 to 10



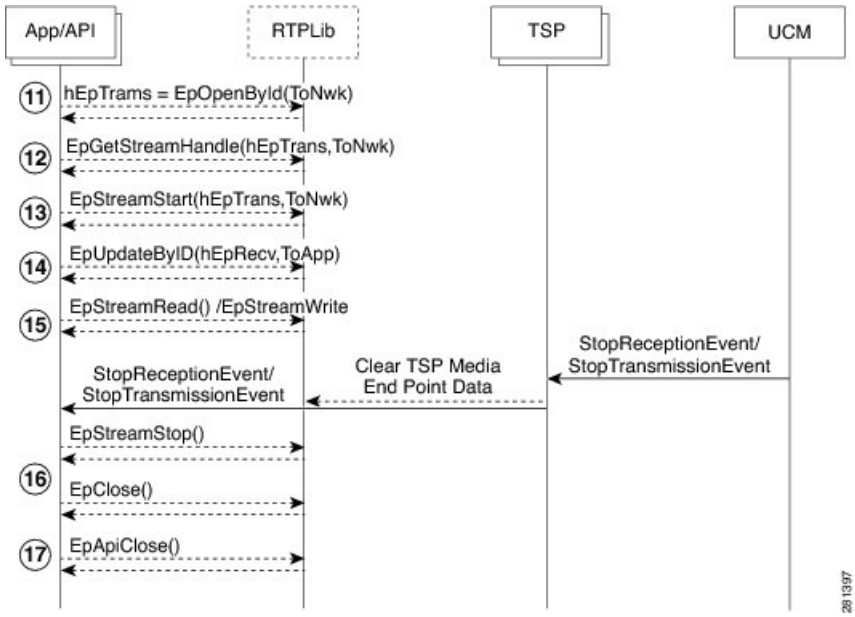
7. Create Media End Point for receiving the data.

8. Get stream handle for Media End Point created for receiving the data.
9. Open the port and start buffering the data on the receiving port.
10. Start monitoring Media Events.

    *** hEpRecv = StreamHandle for Receiving Stream

    The message flow in the following figure is described in steps 11 to 17.

Figure 7: Application Message Flow for Early Offer Call — Steps 11 to 17



11. Create Media End Point for transmitting the data.
12. Get out stream handle.
13. Start data streaming.
14. Update the opened Media End Point with CODEC and other information available in Media Event.
15. Receive/transmit data.
16. Stop data streaming and close end point.
17. Close EpAPI before exiting program.

    *** hEpTrans = StreamHandle for Transmitting Stream

Note

- For the IPV6 Registered port, GetPort Notification is not reported to the application. For the Dual Mode CTI Port or Route Point the Early Offer is supported with IPV4 addresses capability.
- For the Statically Registered CTI port with Legacy Wave driver, the Early Offer feature is not supported and TSP reports error if new LineDevspecific is requested to enable Feature Support.
- For the Statically Registered CTI port with New Cisco Wave Driver/User control Registered CTI Port, New Get IP and Port Notification applications has to open the port that is assigned for the CTI Port.
- On an Early Offer support registered CTI Port, the applications must send the RTP information about new notification and must not set the RTP information on OpenLogical Channel Notification. If set, it is failed by CTI and an acknowledgement is reported to the application.
- When IPv6 support is added, the application receives GetPort Notification twice (one for Ipv4 and one for Ipv6 address) for dual mode device. When a call is answered, the application can close the unused port based on IPAddressingType in Open Logical Channel Notification.

Note To support this feature, the application must negotiate line extension 0x000B0000 or above.

Interface Changes

See Early Offer, Enable Feature, Get IP and Port Event, Set Status Messages, and Open Logical Channel Events.

Message Sequences

See Early Offer.

Backward Compatibility

This feature is backward compatible.

End-to-End Call Trace

End-to-End Call Trace allows tracing of calls that traverse multiple Cisco voice products, such as Unified Communications Manager, Cisco IOS Gateway, and Cisco Call Center products.

A new tool called System Call Tracing tool is developed to collect call records from all voice platforms to trace specific calls and to troubleshoot call failure or other issues.

This tool uses the calling party number, called party number, and time stamp to find at least one call record from any voice product that it can access. From that single call record, System Call Tracing tool traces the call on all the voice products it has traversed.

Interface Changes

LINECALLINFO::DEVSPECIFIC is modified to include TSP Unique Call Reference ID. For more information, see LINECALLINFO.

Message Sequences

See End-To-End Call Trace.

Backward Compatibility

New extension 0x000A0000 is added to maintain backward compatibility.

EnergyWise DeepSleep Mode Support

This feature allows the phone to participate in an EnergyWise enabled system. The phone reports its power usage to a EnergyWise compliant switch to allow the tracking and control of power within the customer premise. The phone provides alternate reduced power modes including an extremely low, off mode. The Unified Communications Manager administrator configures and exclusively manages the phones power state through vendor specific configuration on the Cisco Unified CM Admin pages.

When the phone turns off power after negotiation with an EnergyWise switch, it unregisters from Cisco Unified CM and enters Deep Sleep/PowerSavePlus mode. Phones automatically re-register back with the Cisco Unified CM once the Deep Sleep mode configured PowerON time is reached.

However, for Cisco Unified IP Phones Series 9900 and 6900 phones, press the select key on the phone to wake up the phone from the Deep Sleep/PowerSavePlus mode, but there is no way to register Cisco Unified IP Phones 7900 Series phones back to the Cisco Unified CM during Deep Sleep. This is the limitation for the Cisco Unified IP Phones 7900 Series phones. You can configure Deep Sleep mode on the Device page of the Cisco Unified CM. Configure Deep Sleep mode for the phones at least 10 minutes before the actual power off time to allow the information to synchronize between the switch and the phone.

Power off idle timer enables only in the case when there is physical interaction on the phone. For example if there is a call on the EnergyWise configured phone during the deep sleep time and the user tries to disconnect the call from the application, then the power off idle timer defaults to 10 minutes but if the user disconnects the call manually from the phone, then the power off idle timer takes the value configured on the Cisco Unified CM device page.

TAPI provides the PHONE_STATE message with dwparam1 = PHONESTATE_SUSPEND and EnergyWisePowerSavePlus reason in dwParam2 when the phone unregisters as it enters DeepSleep, and if the phone successfully negotiates with the appropriate extension version 0x000B0000 or higher.

TAPI provides the LINE_LINEDEVSTATE message with dwparam1 = LINEDEVSTATE_OUTOFSERVICE and EnergyWisePowerSavePlus reason in dwparam2 when the phone unregisters as it enters DeepSleep, and if the phone successfully negotiates with the appropriate extension version 0x000B0000 or higher.

As part of this feature TAPI exposes all out of service reason codes in the PHONESTATE_SUSPEND and LINEDEVSTATE_OUTOFSERVICE in dwParam2 when the phone unregisters, and if the phone successfully negotiates with the appropriate extension version 0x000B0000 or higher.

TAPI defines a new enum CiscoLineDevStateOutOfServiceReason in CiscoLineDevSpecificMsg.h

And enum CiscoPhoneStateOutOfServiceReason in CiscoPhoneDevSpecificMsg.h

Interface Changes

```
New Enum under CiscoLineDevSpecificMsg.h
 enum CiscoLineDevStateOutOfServiceReason
{
```

```
    CiscoLineDevStateOutOfServiceReason_Unknown = 0x00000000,
    CiscoLineDevStateOutOfServiceReason_CallManagerFailure = 0x00000001,
    CiscoLineDevStateOutOfServiceReason_ReHomeToHigherPriorityCM = 0x00000002,
    CiscoLineDevStateOutOfServiceReason_NoCallManagerAvailable = 0x00000003,
    CiscoLineDevStateOutOfServiceReason_DeviceFailure = 0x00000004,
    CiscoLineDevStateOutOfServiceReason_DeviceUnregistered = 0x00000005,
    CiscoLineDevStateOutOfServiceReason_EnergyWisePowerSavePlus = 0x00000006,
    CiscoLineDevStateOutOfServiceReason_CtiLinkFailure = 0x00000101
};

New Enum under CiscoPhoneDevSpecificMsg.h
enum CiscoPhoneStateOutOfServiceReason
{
    CiscoPhoneStateOutOfServiceReason_Unknown = 0x00000000,
    CiscoPhoneStateOutOfServiceReason_CallManagerFailure = 0x00000001,
    CiscoPhoneStateOutOfServiceReason_ReHomeToHigherPriorityCM = 0x00000002,
    CiscoPhoneStateOutOfServiceReason_NoCallManagerAvailable = 0x00000003,
    CiscoPhoneStateOutOfServiceReason_DeviceFailure = 0x00000004,
    CiscoPhoneStateOutOfServiceReason_DeviceUnregistered = 0x00000005,
    CiscoPhoneStateOutOfServiceReason_EnergyWisePowerSavePlus = 0x00000006,
    CiscoPhoneStateOutOfServiceReason_CtiLinkFailure = 0x00000101
};
```

Message Sequences

See EnergyWise Deep Sleep Mode Use Cases

Backwards Compatibility

This feature is backward compatible.

Extension Mobility

Extension Mobility, a Unified Communications Manager feature, allows a user to log in and log out of a phone. Cisco Extension Mobility loads a user Device Profile (including line, speed dial numbers, and so on) onto the phone when the user logs in.

Cisco Unified TSP recognizes a user who is logged into a device as the Cisco Unified TSP User.

Using Unified Communications Manager, you can associate a list of controlled devices with a user.

When the Cisco Unified TSP user logs into the device, the system places the lines that are listed in the user Cisco Extension Mobility profile on the phone device and removes lines that were previously on the phone. If the device is not in the controlled device list for the Cisco Unified TSP User, the application receives a PHONE_CREATE or LINE_CREATE message. If the device is in the controlled list, the application receives a LINE_CREATE message for the added line and a LINE_REMOVE message for the removed line.

When the user logs out, the original lines get restored. For a non-controlled device, the application perceives a PHONE_REMOVE or LINE_REMOVE message. For a controlled device, it perceives a LINE_CREATE message for an added line and a LINE_REMOVE message for a removed line.

Extension Mobility Cross Cluster

Extension Mobility Cross Cluster allows users provisioned in one cluster to log in to an IP phone in another cluster.

For this feature, Extension Mobility profile can be added to the control list in addition to the devices. When this profile is added to the control list and an Extension Mobility Cross Cluster user logs in to or logs out of a device within a cluster or either across the cluster, Cisco TSP notifies the application with required Phone Create/Line_Create and Phone_Remove/Line_Remove events.

Interface Changes

None

Message Sequences

See Extension Mobility Cross Cluster.

Backward Compatibility

This feature is backward compatible.

Extension Mobility Memory Optimization Option

The Extension Mobility (EM) feature supports Cisco Unified TSP to use TAPI LINE_CREATE / LINE_REMOVE mechanism to dynamically create and remove line devices resulting from EM login or logout. TAPI, by design, does not remove a device dynamically and marks the device as 'not available'. It remains in memory until the provider is shutdown. As a result, when the EM feature is used, memory utilization grows over time (during login/logout operations) until the memory is exhausted. In many cases, the only workaround is to restart the telephony service or reboot a TAPI client machine.

The EM Memory Optimization Option feature is intended to minimize the usage of LINE_CREATE / LINE_REMOVE in EM-related scenarios by reusing TAPI device IDs for lines from different EM profiles loaded on the same IP Phone. Lines with the same index in different EM profiles share the same TAPI line device ID.

The feature can be enabled or disabled by using the registry settings. By default, the feature is disabled so that the existing applications are not affected.

If the feature is enabled, LINE_CREATE messages are used by Cisco Unified TSP only for the first time when an EM profile is loaded on a particular IP Phone. After the EM line is created, it is not removed with LINE_REMOVE. It is instead placed in the Inactive state when EM logout occurs. Operations cannot be performed when a device is in the Inactive state and the LINEERR_DEVICE_INACTIVE error returns if an operation is invoked.

The line is reactivated when an EM profile is reloaded on the IP Phone as a result of a new EM login. Along with the line reactivation notification, the application is also notified that line device capabilities have changed. The Other-Device State Notification feature is utilized for delivering active, inactive, and capability change messages to an application. For more information, see Other-Device State Notification, on page 56.

Note The EM Memory Optimization Option feature intends to minimize the usage of LINE_CREATE/LINE_REMOVE messages. Even if the feature is enabled, an application can still receive the LINE_CREATE and LINE_REMOVE messages in some scenarios. So the application has to be written in a way that it can handle the existing LINE_CREATE/LINE_REMOVE events along with the new LineActive/LineInactive notifications.

Installation/Configuration Change

This feature can be enabled or disabled using the registry settings. The registry settings are stored in the EMOptions registry entry that is created during the Cisco TSP installation or upgrade. There is only one EMOptions entry in the Cisco TSP registry settings which applies to all Cisco TSP instances on the box.

The EMOptions registry settings must be changed manually. There is no Cisco TSP configuration interface to modify the setting and no possibility to change the feature behavior dynamically. Cisco TSP must be restarted for the modified setting to take effect.

A corresponding parameter is also available for a silent Cisco TSP install. This allows enabling or disabling of the feature at the time of the silent install/upgrade process.

Interface Changes

CiscoLineDevStateCloseReason provides details to the LINEDEVSTATE_CLOSE state and is passed to the application as dwParam2 in the LINE_LINEDEVSTATE message.

```
enum CiscoLineDevStateCloseReason
{
CiscoLineDevStateCloseReason_Unknown = 0,
CiscoLineDevStateCloseReason_LineNotAvailable,
CiscoLineDevStateCloseReason_EMActivity
};
```

Message Sequences

See Extension Mobility Memory Optimization Option.

Backward Compatibility

This feature can be enabled or disabled using the registry settings. By default, this feature has been disabled so that the existing applications are not affected.

External Call Control

External Call Control enables Unified Communications Manager to route calls based on enterprise policies and presence-based routing rules of individual users. When External Call Control is enabled, Unified Communications Manager queries the designated web services hosting the enterprise policies or user rules and routes the calls based on the routing decisions returned.

As TSP receives the expected unmodified Directory Number or partition in all of the party fields, most scenarios remain unaffected by the External Call Control feature. TSP passes these unmodified fields in the PartyId fields to TAPI. Since it is also possible for the External Call Control feature to change the modified calling and called parties, TSP passes this in the existing modified fields of the devSpecific part of lineCallInfo.

With the changes made by CTI for the External Call Control feature, TSP also supports Translation Patterns. Calls going through translation patterns are supported by the TSP and if these calls are involved in a conference, the correct number of CONFERENCE calls shall be created and maintain for the duration of the conference.

Interface Changes

New CtiReasonExternalCallControl (42) in the ExtendedCallReason field in the devSpecific part of lineCallInfo for some intercept scenarios.

New dev specific error, LINEERR_OPERATION_FAIL_CHAPERONE_DEVICE, is returned in LINE_REPLY for any request that is rejected for a device which is involved in a chaperone call except for lineSetupConference/lineAddToConference, lineDevSpecific(SLDST_START_CALL_RECORDING), and lineDrop requests.

In the CallAttributeBitMask field, LINECALLINFO::DEVSPECIFIC is modified to include a new bit mask, TSPCallAttribute_ChaperoneCall. For more information, see Details.

Message Sequences

See External Call Control.

Backward Compatibility

This feature is backward compatible.

Note As TSP did not support Translation Patterns before this release, the support of Translation Patterns is considered backward compatible even though there is a change in the CallInfo for calls using Translation Patterns.

FIPS Compliance

Federal Information Processing Standards (FIPS) are publicly announced standards developed by the United States federal government for use in computer systems by all non-military government agencies and government contractors. The FIPS 140-2 requirements have been defined jointly by the American NIST (National Institute for Standards and Technology) and the Canadian CSEC (Communications Security Establishment of Canada).

FIPS compliance support has been added to Unified Communications Manager. This mode can be enabled or disabled using CallManager Administration Command Line Interface.

From CiscoTSP, FIPS Compliance is supported by upgrading to FIPS 140-2 validated OpenSSL Version, “FIPS capable OpenSSL”, library which is used for setting TLS connection with CTI Manager or Unified Communications Manager. FIPS mode on CiscoTSP can be updated dynamically. Currently, CiscoTSP enables FIPS Mode and depends on FIPS Mode of the Unified Communications Manager.

Interface Changes

No interface changes.

Message Sequences

No impact on end user.

Backward Compatibility

This feature is backwards compatible.

Conference Changes

Forced Authorization Code and Client Matter Code

Cisco Unified TSP supports and interacts with two Unified Communications Manager features: Forced Authorization Code (FAC) and Client Matter Code (CMC). The FAC feature lets the System Administrator require users to enter an authorization code to reach certain dialed numbers. The CMC feature lets the System Administrator require users to enter a client matter code to reach certain dialed numbers.

The system alerts a user of a phone that a FAC or CMC must be entered by sending a “ZipZip” tone to the phone that the phone in turn plays to the user. Cisco Unified TSP will send a new LINE_DEVSPECIFIC event to the application whenever the application should play a “ZipZip” tone. Applications can use this event to indicate when a FAC or CMC is required. For an application to start receiving the new LINE_DEVSPECIFIC event, it must perform the following steps:

1. lineOpen with dwExtVersion set to 0x00050000 or higher
2. lineDevSpecific – Set Status Messages to turn on the Call Tone Changed device specific events

The application can enter the FAC or CMC code with the lineDial() API. Applications can enter the code in its entirety or one digit at a time. An application may also enter the FAC and CMC code in the same string as long as they are separated by a “#” character and also ended with a “#” character. The optional “#” character at the end only serves to indicate dialing is complete.

If an application does a lineRedirect() or a lineBlindTransfer() to a destination that requires a FAC or CMC, Cisco Unified TSP returns an error. The error that Cisco Unified TSP returns indicates whether a FAC, a CMC, or both are required. Cisco Unified TSP supports two new lineDevSpecific() functions, one for Redirect and one for BlindTransfer, that allows an application to enter a FAC or CMC, or both, when a call gets redirected or blind transferred.

Forwarding

Cisco Unified TSP now provides added support for the lineForward() request to set and clear ForwardAll information on a line. This will allow TAPI applications to set the Call Forward All setting for a particular line device. Activating this feature will allow users to set the call forwarding Unconditionally to a forward destination.

Cisco Unified TSP sends LINE_ADDRESSSTATE messages when lineForward() requests successfully complete. These events also get sent when call forward indications are obtained from the CTI, indicating that a change in forward status has been received from a third party, such as Unified Communications Manager Administration or another application setting call forward all.

Gateway Recording

Unified Communications Manager has been providing a recording solution since release 6.0. In previous releases, the call recording was phone-based. The Cisco IP phones are used to fork, or direct, the two media streams of the agent-customer call to the recorder.

However, for call scenarios where the devices involved do not directly register with Unified Communications Manager, phone-based recording is not possible. This situation excludes the calls handled by the mobile agents from being recorded. In addition, the recording of mobile calls becomes increasingly mandatory by regulations in different jurisdictions, or becomes the essential business requirement for call centers or enterprises. These requirements call for a recording solution that does not rely on media forking from the endpoints.

The enhancements for the recording solution allow an external call that goes through a Cisco voice gateway to be recorded by having the voice gateway direct the two media streams to a voice recorder. This solution uses the existing CCiscoLineDevSpecificStartCallRecording and CCiscoLineDevSpecificStopCallRecording APIs to start and stop a user control recording session. Automatic recording is also configured on the line. There is an additional option to specify the voice gateway, or the IP phone as the prefered recording resource on a particular line.

In addition, the CTI Remote Device that was introduced in Unified Communications Manager Release 9.0 now supports recording through a gateway enabled for recording. If there is a recording gateway between the CTI Remote Device and the remote-destination where the call was routed, the recording can be started at the CTI Remote Device. If there is a recording gateway between the caller and the CTI Remote Device, the recording can also be started at the CTI Remote Device. You can also configure the line(s) on the CTI Remote Device to support Automatic or Selective Recording. The CTI Remote Device can only use a gateway to route the media.

CTI port calls can be captured using the Network Recording feature available in Unified Communications Manager Release 10.0(1). The call media must pass through at least one recording-enabled gateway to be recorded. A typical use case: An external call to a CTI Port softphone. The Recording Media Source for the CTI Port is always gateway-preferred.

Interface Changes

- Recording Failure Event
- LINECALLINFO
- LINEDEVCAPS

Message Sequences

Gateway Recording

Backward Compatibility

Due to a design change to support Gateway Recording, the same recording may be stopped and restarted due to a feature invocation that was not seen in prior releases. Previously, this stop and start of the recording was only seen when the call being recorded is put on hold and then resume. In this release, this can happen when the party on the other end of the call being recorded changes. Take the following example:

Action	Pre-10.0	10.0
A calls B and B answers		
TSP application issues CCiscoLineDevSpecificStartCallRecording at A.	App receives SLDSMT_RECORDING_STARTED event at A	App receives SLDSMT_RECORDING_STARTED event at A

Action	Pre-10.0	10.0
B transfers the call to C and C answers	No event is received at A	App receives SLDSMT_RECORDING_ENDED event at A App receives SLDSMT_RECORDING_STARTED event at A

Applications do not need to start the recording again but must handle these extra events.

Hold Reversion

The Hold Reversion feature allows a holding party to be notified about the HELD call after the hold reversion duration times out. The holding party gets audio and visible hold reversion notifications. The application can set the hold reversion event flag to receive the hold reversion notification from Cisco TSP. CallInfo and CallState of the call remains unchanged when a hold reversion event occurs and the LineCallDevSpecific event is sent to the application indicating the hold reversion if the application has enabled the hold reversion event flag.

Hunt List

CiscoTSP supports lines and their devices included in the Hunt List and provides appropriate information for applications to understand that the call is offered through a Hunt Pilot. Hunt List can include more than one Line Group and each Line Group may have different call distribution algorithms. Irrespective of the algorithm used in the Line Groups, CiscoTSP provides consistent information to applications.

When call offered on the line is routed through Hunt Pilot, CiscoTSP provides Hunt Pilot Directory Number or Partition in LINECALLINFO::DEVSPECIFIC for Caller, Called, or Connected IDs. However, there will be no Hunt Pilot information for Redirection and Redirecting ID. Hunt Pilot name is not passed to LINECALLINFO::DEVSPECIFIC.

There is no separate LineDevSpecific event to report the Hunt Pilot information, however, it is reported by existing event when caller, called, or connected id changes.

Hunt List can also interact with Call Pickup feature. However, Hunt List broadcast feature is not supported while interacting with Call Pickup feature. In this case, there will be no Pickup notification for the broadcasted call. CTI or Unified Communications Manager fails the pickup request if the application tries to pickup a broadcast call in Hunt List.

When call is routed to Hunt List and offering is accepted by Line Group member, the called information becomes Hunt Pilot's information. After Line Group member answers the call, connected ID will shows the actual Line Group member information with Hunt Pilot information in LINECALLINFO::DEVSPECIFIC. In this case, both called and connected ID contains Hunt Pilot information.

In case of a conference, Hunt Pilot Directory Number or Partition is presented if connected party's primary call is routed through Hunt List.

Note To support this feature, the application must negotiate line extension 0x00A0000 or above.

Interface Changes

LINECALLINFO.

Message Sequences

See Hunt List.

Backward Compatibility

This feature is backward compatible.

Hunt Pilot Connected Number

In Unified Communications Manager 9.0, the support for hunt pilots is enhanced to expose the huntmember which has answered the call as the called party in a call involving a hunt pilot at the calling side. With this enhancement, when a user calls a hunt pilot HP and the call is answered by the hunt list Line group member LG1, TAPI exposes DNof the HuntMember as the ModifiedConnectedParty DN under devspecific part of linecallinfo under Call Party Normalization info structure.

When this feature is disabled, the modifiedconnectedParty exposed is the HuntPilotDN.

The HuntPilot Information is available in the devspecific part of linecallinfo(under HuntPilotInfo structure). There is no change in Huntpilot information for call scenarios involving huntpilot, when this feature is enabled from the information exposed when this is feature is disabled.

Interface Changes

Not applicable.

Message Sequences

See Hunt Pilot Connected Number Feature.

Backward Compatibility

This feature is backward compatible. It can be enabled on Huntpilot page **Display Line Group Member DN as Connected Party**. By default, this feature is disabled so that the existing applications are not affected.

Hunt Group Login Status

This feature allows TSP applications to log a controlled device in and out of a hunt group. In addition, TSP applications can query a device for the hunt group login status.

After the TSP application successfully logs in or logs out of a hunt group, the TSP application sends a phone event to notify associated applications of the updated login status. If the request to change the status does not actually result in a status change (for example, if TSP tries to log a device into a hunt group, but the device was already logged in to that group) no notification is sent.

Interface Updates

To support this feature, two interfaces have been updated:

- To log in and log out of hunt groups, the CCiscoPhoneDevSpecificSetHuntGroupLoginStatus extension has been added to the existing CCiscoPhoneDevSpecific interface. The new extension has an enum mHuntGroupLoginStatus.
- To query the hunt group login status, a new extension Cisco_LineDevCaps_Ext000E0000 has been added to the existing LineGetDevCapsinterface. The new extension has a member variable of DeviceHuntGroupLoginStatus.

Note To log in and out of hunt groups, the application must negotiate line extension 0x000E0000 or above. The new line extension is added as a part of this feature.

To make use of this feature, he application must open a TAPI phone device with minimum extension version 0x00030000.

Error Description

Hunt group logins are not supported with the following device types. Login requests return the LINEERR_OPERATIONUNAVAIL error:

- DeviceTapiRoutePoint
- CtiRemoteDevice
- CtiSparkRemoteDevice

The range of enum values for Hunt Group Login status must fall within the range of 0 to 2. Otherwise, the request returns LINEERR_INVALPARAM.

If the request to set the Hunt Group Login Status is in process and a second request is sent, the second request returns the LINEERR_PENDING_REQUEST error.

Interface Changes

Hunt Group Login Status

Message Sequences

Hunt Group Login Status

Backward Compatibility

none

Intercom

The Intercom feature allows one user to call another user and have the call automatically answered with one-way media from the caller to the called party, regardless of whether the called party is busy or idle.

To ensure that no accidental voice of the called party is sent back to the caller, Unified Communications Manager implements a ‘whisper’ intercom, which means that only one-way audio from the caller is connected, but not audio from the called party. The called party must manually press a key to talk back to the caller. A

zip-zip (auto-answer) tone will play to the called party before the party can hear the voice of the caller. For intercom users to know whether the intercom is using one-way or two-way audio, the lamp for both intercom buttons appears colored amber for a one-way whisper Intercom and green for two-way audio. For TSP applications, only one RTP event occurs for the monitored party: either the intercom originator or the intercom destination, with the call state as whisper, in the case of a one-way whisper intercom.

TSP exposes the Intercom line indication and Intercom Speed Dial information in DevSpecific of LineDevCap. The application can retrieve the information by issuing LineGetDevCaps. In the DevSpecific portion, TSP provides information that indicates (DevSpecificFlag = LINEDEVCAPSDEVSPECIFIC_INTERCOMDN) whether this is the Intercom line. You can retrieve the Intercom speed dial information in the DevSpecific portion after the partition field.

If a CTI port is used for the Intercom, the application should open the CTI port with dynamic media termination. TSP returns LINEERR_OPERATIONUNAVAIL if the Intercom line is opened with static media termination.

Note You cannot use CTI Route Point for the Intercom feature.

The administrator can configure the speed dial and label options from Unified Communications Manager Administration. However, the application can issue CciscoLineSetIntercomSpeeddial with SLDST_LINE_SET_INTERCOM_SPEEDDIAL to set or reset SpeedDial and Label for the intercom line. The Application setting will overwrite the administrator setting that is configured in the database. Unified Communications Manager uses the application setting to make the intercom call until the line is closed or until the application resets it. In the case of a Communications Manager or CTIManager failover, CTIManager or Cisco TSP resets the speed dial setting of the previous application. If the application restarts, the application must reset the speed dial setting; otherwise, Unified Communications Manager will use the database setting to make the intercom call. In any case, if resetting of the speed dial or label fails, the system sends a LINE_DEVSPECIFIC event to indicate the failure. When the application wants to release the application setting and have the speed dial setting revert to the database setting, the application can call CciscoLineSetIntercomSpeeddial with a NULL value for SpeedDial and Label.

If the speed dial setting is changed, whether due to a change in the database or because the application overwrites the setting, the system generates a LineDevSpecific event to indicate the change. However, the application needs to call CCiscoLineDevSpecificSetStatusMsgs with DEVSPECIFIC_SPEEDDIAL_CHANGED to receive this notification. After receiving the notification, the application can call LineGetDevCaps to find out the current settings of speed dial and label.

Users can initiate an intercom call by pressing the Intercom button at the originator or by issuing a LineMakeCall with a NULL destination if Speedial/Label is configured on the intercom line. Otherwise, LineMakeCall should have a valid Intercom DN.

For an intercom call, a CallAttribute field in LINECALLINFODEVSPECIFIC indicates whether the call is for the intercom originator or the intercom target.

After the intercom call is established, the system sends a zip-zip tone event to the application as a tone-changed event.

Users can invoke a TalkBack at the destination in two ways:

- By pressing the intercom button
- By issuing CciscoLineIntercomTalkback with SLDST_LINE _INTERCOM_TALKBACK

TSP reports the Whisper call state in the extended call state bit as CLDSMT_CALL_WHISPER_STATE. If the call is being put on hold because the destination is answering an intercom call by using talk back, the system reports the call reason CtiReasonTalkBack in the extended call reason field for the held call.

The application cannot set line features, such as set call forwarding and set message waiting, other than to initiate the intercom call, drop the intercom call, or talk back. After the intercom call is established, the system limits call features for the intercom call. For the originator, only LINECALLFEATURE_DROP is allowed. For the destination, the system supports only the LINECALLFEATURE_DROP and TalkBack features for the whisper intercom call. After the intercom call becomes two-audio after the destination initiates talk back, the system allows only LINECALLFEATURE_DROP at the destination.

Speed dial labels support unicode.

IPv6

The IPv6 support feature enables IPv6 capabilities in a Unified Communications Manager network. IPv6 increases the number of addresses available for network devices. TAPI can connect to Unified CM with IPv6 support if the IPv6 Support feature is enabled on Unified CM. IPv6 enhancements include the following:

- Provides the IPv6 address of the calling party to the called party in the Devspecific part of LINECALLINFO.
- Support to register a CTI port or a route point with an IPv6 address. The RTP destination address also contains IPv6 addresses if the same is involved in media establishment.

The TSP user interface includes the primary and backup CTI Manager address and a flag that indicates the preference of user while connecting to the CTI Manager. CTI ports and route points can be registered with IPv4, IPv6, or both.

The following new CiscoLineDevSpecific functions allow the application to specify IP address mode and IPv6 address before opening CTI port and route point:

- CciscoLineDevSpecificSetIPv6AddressAndMode
- CciscoLineDevSpecificSetRTPParamsForCallIPv6

For dynamic port registration, on receiving the SLDSMT_OPEN_LOGICAL_CHANNEL event, the CciscoLineDevSpecificSetRTPParamsForCallIPv6 allows the application to provide IPv6 information for the call.

Interface Changes

See Set IPv6 Address and Mode.

Message Sequences

See IPv6 Use Cases.

Backward Compatibility

This feature is backward compatible. The 0x00090000 extension must be negotiated to use this feature.

Transfer Changes

Join

In Unified Communications Manager, the Join softkey joins all the parties of established calls (at least two) into one conference call. The Join feature does not initiate a consultation call and does not put the active call on hold. It also can include more than two calls, which results in a call with more than three parties.

Cisco Unified TSP exposes the Join feature as a new device-specific function that is known as lineDevSpecific() – Join. Applications can apply this function to two or more calls that are already in the established state. This also means that the two calls do not need to be set up initially by using the lineSetupConference() or linePrepareAddToConference() functions.

Cisco Unified TSP also supports the lineCompleteTransfer() function with dwTransferMode = Conference. This function allows a TAPI application to join all the parties of two, and only two, established calls into one conference call.

Cisco Unified TSP also supports the lineAddToConference() function to join a call to an existing conference call that is in the ONHOLD state.

A feature interaction issue involves Join, Shared Lines, and the Maximum Number of Calls. The issue occurs when you have two shared lines and the maximum number of calls on one line is less than the maximum number of calls on the other line.

For example, in a scenario where one shared line, A, has a maximum number of calls set to 5 and another shared line, A', has a maximum number of calls set to 2, the scenario involves the following steps:

A calls B. B answers. A puts the call on hold.

C calls A. A answers. C puts the call on hold.

At this point, line A has two calls in the ONHOLD state, and line A' has two calls in the CONNECTED_INACTIVE state.

D calls A. A answers.

At this point, the system presents the call to A, but not to A'. This happens because the maximum calls for A' specifies 2.

A performs a Join operation either through the phone or by using the lineDevSpecific – Join API to join all the parties in the conference. It uses the call between A and D as the primary call of the Join operation.

Because the call between A and D was used as the primary call of the Join, the system does not present the ensuing conference call to A'. Both calls on A' will go to the IDLE state. As the end result, A' will not see the conference call that exists on A.

Join Across Lines (SCCP)

This feature allows two or more calls on different lines of the same device to be joined through the join operation. Applications can use the existing join API to perform the task. When the join across line happens, the consultation call on the different line on which the survival call does not reside will get cleared, and a

CONFERENCED call that represents the consultation call will be created on the primary line where conference parent is created. This feature should have no impact when multiple calls are joined on the same line.

This feature is supported on SCCP devices that can be controlled by CTI. In addition, this feature also supports chaining of conference calls on different lines on the same device. Also, a join across line can be performed on a non-controller line; that is, the original conference controller was on a different device then where the join is being performed.

Note This feature returns an error if one of the lines that are involved in the Join Across Lines is an intercom line.

Backwards Compatibility

This feature is backward compatible.

Join Across Lines (SIP)

This feature allows two or more calls on different lines of the same device to be joined by using the join operation. Applications can use the existing join API to perform the task. When the join across line happens, the consultation call on the different line on which the survival call does not reside will get cleared, and a CONFERENCED call that represents the consultation call will get created on the primary line where conference parent is created. This feature should have no impact when multiple calls are joined on the same line.

This feature is supported both on SCCP and SIP devices that can be controlled by CTI. In addition, this feature also supports chaining of conference calls on different lines on the same device. Also, a join across line can be performed on a non-controller (the original conference controller was on a different device then where the join is being performed) line.

This feature returns an error if one of the lines involved in the Join Across Lines is an intercom line.

Interface Changes

None.

Message Sequences

See Join Across Lines.

Backwards Compatibility

This feature is backward compatible.

Line-Side Phones That Run SIP

TSP supports controlling and monitoring of TNP-based phones that are running SIP. Existing phones (7960 and 7940) that are running SIP cannot be controlled or monitored by the TSP and should not get included in the control list. Though the general behavior of a phone that is running is similar to a phone that is running SCCP not all TSP features get supported for phones that are running SIP.

CCiscoPhoneDevSpecificDataPassThrough functionality does not support on phones that are running SIP configured with UDP transport due to UDP limitations. Phones that are running SIP must be configured to use TCP (default) if the CCiscoPhoneDevSpecificDataPassThrough functionality is needed.

TSP application registration state for TNP phones that are running SIP with UDP as transport may not remain consistent to the registration state of the phone. TNP phone that are running SIP with UDP as transport may appear to be registered when application reports the devices as out of service. This may happen when CTIManager determines that Unified CM is down and puts the device as out of service, but, because of the inherent delay in UDP to determine the lost connectivity, phone may appear to be in service.

The way applications open devices and lines on phones that are running SIP remains the same as that of phone that is running SCCP. It is the phone that controls when and how long to play reorder tone. When a SIP phone gets a request to play reorder tone, the phone that is running SIP releases the resources from Unified CM and plays reorder tone. The call appears to be IDLE to a TSP application even though reorder tone is being played on the phone. Applications can still receive and initiate calls from the phone even when reorder tone plays on the phone. Because resources have been released on Unified CM, this call does not count against the busy trigger and maximum number of call counters.

When consult call scenario is invoked on the SIP, the order of new call event (for consult call) and on hold call state change event (for original call).

Localization Infrastructure Changes

Beginning with Release 7.0(1), the TSP localization is automated. The TSP UI can download the new and updated locale files directly from a configured TFTP server location. As a result of the download functionality, Cisco TSP install supports only the English language during the installation.

During installation, the user inputs the TFTP server IP address. When the user opens the TSP interface for the **first time**, the TSP interface automatically downloads the locale files from the configured TFTP server and extracts those files to the resources directory. The languages tab in the TSP preferences UI also provides functionality to update the locale files.

Note To upgrade from Unified Communications Manager, Release 6.0(1) TSP to Cisco Unified Communications Manager, Release 7.0(1) TSP, you must ensure that Release 6.0(1) TSP was installed by using English. If Release 6.0(1) TSP is installed using any language other than English and if the user upgrades to Release 7.0(1) TSP, then the user must manually uninstall Release 6.0(1) TSP from Add/Remove programs in control panel and then perform a fresh install of Release 7.0(1) TSP.

Interface Changes

None.

Message Sequences

None.

Backward Compatibility

Only English locale is packaged in Cisco TSP installer. The TSP UI downloads the locale files from the configured TFTP server. The end user can select the required and supported locale after the installation.

Logical Partitioning

The Logical Partitioning feature restricts VoIP to PSTN calls and vice versa, based on the logical partitioning policy. Any request that interconnects a VOIP call to a PSTN call or vice versa in two different geographical locations fails and the error code is sent back to the applications.

The device, device pool, trunk, and gateway pages now provide configuration to select geo-location values and construction rules for geo-location strings.

A new enterprise parameter has been added for this feature with the following values:

- Name: Logical partitioning enabled
- Values: True or False
- Default: False

A new error code has been added for this feature: LINEERR_INVALID_CALL_PARTITIONING_POLICY 0xC000000C

Interface Changes

There are no interface changes for this feature.

Message Sequences

See Logical Partitioning.

Backward Compatibility

This feature is backward compatible. To maintain earlier behavior, set the logical partitioning enabled parameter to **False**.

Message Waiting Indicator Enhancement

The Message Waiting Indicator (MWI) feature enhancement enables the application to display the following information on the supported phones:

- Total number of new voice messages (normal and high priority messages)
- Total number of old voice messages (normal and high priority messages)
- Number of new high priority voice messages
- Number of old high priority voice messages
- Total number of new fax messages (normal and high priority messages)
- Total number of old fax messages (normal and high priority messages)
- Number of new high priority fax messages
- Number of old high priority fax messages

MWI also includes two CCiscoLineDevSpecific subclasses are added to enhance the MWI functionality. Similar to the existing setMessageWaiting operation, one MWI operation sets the summary information for the controlled line, while the another MWI operation sets the message summary information on any line that is reachable by the controlled line, as defined by the configured calling search space of the controlled line.

Interface Changes

See Message Summary and Message Summary Dirn.

Message Sequences

There are no message sequences for this feature.

Backward Compatibility

This feature is backward compatible.

Microsoft Windows Vista

Microsoft Windows Vista operating system supports Cisco TSP client with the following work around:

- Always perform the initial installation of the Cisco TSP and Unified Communications Manager TSP Wave Driver as a fresh install.
- If a secure connection to Unified Communications Manager is used, turn off/disable the Windows Firewall.
- If Unified Communications Manager TSP Wave Driver is used for inbound audio streaming, turn off/disable the Windows Firewall.

If Unified Communications Manager TSP Wave Driver is used for audio streaming, you must disable all other devices in the Sound, Video, and Game Controllers group.

Monitoring Call Park Directory Numbers

Cisco TSP supports monitoring calls on lines that represent Unified Communications Manager Administration Call Park Directory Numbers (Call Park DNs). Cisco TSP uses a device-specific extension in the LINEDEVCAPS structure that enables TAPI applications to differentiate Call Park DN lines from other lines. If an application opens a Call Park DN line, all calls that are parked to the Call Park DN are reported to the application. The application cannot perform any call-control functions on any of the calls at a Call Park DN.

In order to open Call Park DN lines, the Monitor Call Park DNs check box in theUnified Communications Manager Administration for the TSP user must be checked. Otherwise, the application will not see any of the Call Park DN lines upon initialization.

Multiple Calls Per Line Appearance

The following topics describe the conditions of Line Appearance.

Maximum Number of Calls

The maximum number of calls per Line Appearance remains database configurable, which means that the Cisco TSP supports more than two calls per line on Multiple Call Display (MCD) devices. An MCD device comprises a device that can display more than two call instances per DN at any given time. For non-MCD devices, the Cisco TSP supports a maximum of two calls per line. The absolute maximum number of calls per line appearance equals 200.

Busy Trigger

In Cisco Unified CM, a setting, busy trigger, indicates the limit on the number of calls per line appearance before the Cisco Unified CM will reject an incoming call. Be aware that the busy trigger setting is database configurable, per line appearance, per cluster. The busy trigger setting replaces the old call waiting flag per DN. You cannot modify the busy trigger setting using the CiscoTSP.

Call Forward No Answer Timer

Be aware that the Call Forward No Answer timer is database configurable, per DN, per cluster. You cannot configure this timer using the CiscoTSP.

New Cisco Media Driver

Cisco TSP now allows the application to use the new Cisco Media Driver (next generation Wave Driver). Cisco Media Driver provides applications with functions similar to the legacy kernel mode driver, has improved scalability, and supports latest Microsoft operating system releases.

Two additional device classes, ciscowave/in and ciscowave/out, have been introduced to support the new driver. These classes can be used in the lineGetID() to retrieve line-device identifiers for media devices associated with line when the new Cisco Media Driver is used for audio streaming. For more information, see Cisco TSP Media Driver.

Interface Changes

None

Message Sequences

None

Backward Compatibility

This feature is not backward compatible.

Other-Device State Notification

TAPI specification does not have a provision to notify applications regarding state changes of non-opened devices. The Other-Device State Notification feature enhances Cisco TSP with that functionality. Using this feature, an application can assign devices that Cisco TSP should use to notify the application about non-opened device state changes.

Currently, Cisco TSP supports this feature only for line devices. An application can enable the feature on any open line device using a DEVSPECIFIC_OTHER_DEVICE_STATE_NOTIFY flag in CCiscoLineDevSpecificSetStatusMsgs lineDevSpecific request. Cisco TSP then uses this line to deliver other-device state change notifications to the application. Notifications are sent in LINE_LINEDEVSTATE message as follows:

```
LINE_LINEDEVSTATE
hDevice = (DWORD) hLine;
dwCallbackInstance = (DWORD) hCallback;
dwParam1 = (DWORD) LINEDEVSTATE_OTHER;
dwParam2 = (DWORD) CISCO_LINEDEVSTATE_OTHER_REASON_constant;
dwParam3 = (DWORD) dwLineDeviceId;
```

CiscoLineDevStateOtherReason enumeration defines all relevant values of the Param2 in this notification. Param3 contains the TAPI identifier of the line device for which the state has changed.

The other-device state notification is a supplementary mechanism that can be used by other features that provide the application with state change notifications for non-opened devices. For example, EM Memory Optimization uses this feature to notify an application when a line device becomes active or inactive as a result of EM login or logout.

Interface Changes

CiscoLineDevStateOtherReason enumeration type provides details to LINEDEVSTATE_OTHER and is passed to the application as dwParam2 in the LINE_LINEDEVSTATE message

```
enum CiscoLineDevStateOtherReason
{
CiscoLineDevStateOtherReason_Unknown = 0,
CiscoLineDevStateOtherReason_OtherLineInactive,
CiscoLineDevStateOtherReason_OtherLineActive,
CiscoLineDevStateOtherReason_OtherLineCapsChange
};
```

DEVSPECIFIC_OTHER_DEVICE_STATE_NOTIFY flag in Set Status Messages, lineDevSpecific request must be used to enable other-device state notifications. For more information, see Set Status Messages.

Message Sequences

See Extension Mobility Memory Optimization Option.

Backward Compatibility

The other-device state notification is a supplementary feature intended to be used by other features that require to provide the application with state change notifications for non-opened devices. Its backward compatibility should be considered in a context of a specific feature. For more information, see Extension Mobility Memory Optimization Option, on page 41.

Park Monitoring

The Park Monitoring feature allows you to monitor the status of parked calls. This feature improves the user experience of retrieving the parked calls. When TAPI receives a parked call notification, a call representing

the parked call is generated, and the call is set to CONNECTED INACTIVE state. The parked call is set to IDLE when it is retrieved or forwarded to Park Monitoring Forward No Retrieve Destination.

DEVSPECIFIC_PARK_STATUS event is sent when call is parked, reminded, retrieved, and aborted. LineDevSpecific SLDST_SET_STATUS_MESSAGES are enhanced to allow the application to enable/disable DEVSPECIFIC_PARK_STATUS event.

When Cisco TSP receives the LINE_PARK_STATUS event for the newly parked call, Cisco TSP simulates a call for each of the newly parked call using the SubID received from the LINE_PARK_STATUS event, and notifies the application about the new parked call using the LINE_NEWCALL event.

Cisco TSP uses LINE_CALLSTATE event to notify changes in the park status to the application. The park status in the LINE_CALLSTATE event can be one of the following:

- Parked -indicates a call is parked by the TSP monitored Cisco Unified IP phone.
- Retrieved -indicates a previously parked call is retrieved.
- Abandoned -indicates a previously parked call is disconnected while waiting to be retrieved.
- Reminder -indicates the park monitoring reversion timer for the parked call has expired.
- Forwarded -indicates the parked call has been forwarded to the configured Park Monitoring Forward No Retrieve destination, or if the FNR destination is not configured, the call is forwarded back to the parker.

When Cisco TSP receives the LINE_PARK_STATUS event, it maps the existing CALLINFO structure with the fields received from LINE_PARK_STATUS event. The application then retrieves the updated structure by invoking lineGetCallInfo.

The mapping of the fields in the LINE_PARK_STATUS event to the LINECALLINFO structure is as follows:

LINE_PARK_STATUS	LINECALLINFO--	Description
LineHandle	hline	Identifies the line handle to which this message applies
GCID	dwcallid	Identifies the global call handle to which this message applies.
TransactionID	dwRedirectingName	A unique ID that identifies a particular parked call
CallReason	dwReason	Identifies the call reason.
Park Status	dwBearerMode	Parked, retrieved, abandoned, reminder, forwarded -indicates the status of the parked call.
ParkSlotDN	dwCallerID	The park slot DN.
ParkSlotPartition	dwCallerIDName	The partition of the park slot DN.
ParkedPartyDN	dwCalledID	The parked party DN.
ParkedPartyPartition	dwCalledIDName	The partition of the parked party DN.

To maintain the existing behavior of the Park feature for the Cisco Unified IP Phones running SIP, you can set the value of the Park Monitoring Forward No Retrieve Destination timer equal to the existing Call Park Duration timer and leave the Park Monitoring Forward No Retrieve Destination blank.

To override the Park Monitoring feature for the Cisco Unified IP Phones running SIP, turn off the DEVSPECIFIC_PARK_STATUS message flag by using the lineDevSpecific SLDST_SET_STATUS_MESSAGES request.

Interface Changes

See Set Status Messages.

Message Sequences

See Park Monitoring.

Backward Compatibility

This feature is backward compatible.

Partition

The CiscoTSP support of this feature will provide Partition support information. Prior to release 5.1, CiscoTSP only reported partial information about the DNs to the applications in that it would report the numbers assigned but not the information about the partitions with which they were associated.

Thus, if a phone has two lines that are configured with same DNs – one in Partition P1 and the other in P2, a TSP application would cannot distinguish between these two and consequently open only one line of these two.

CiscoTSP provides the partition information about all DNs to the applications. Thus, the preceding limitation gets overcome and applications can distinguish between and open two lines on a device, which share the same DN but belong to different partitions.

TSP applications can query for LINEDEVCAPS where the partition information is stored in the devSpecific portion of the structure. The application will receive the partition information for the CallerID, CalledID, ConnectedID, RedirectionID, and RedirectingID in a call. This is provided as a part of DevSpecific Portion of the LINECALLINFO structure.

Also, the partition information of the Call Park DN at which the call was parked is also sent to the applications. The value of the partition information is sent to applications in ASCII format.

Note

Password Expiry Notification

The password expiry notification notifies the user about the password expiry date and provides the specific reason for the initialization failure if the password is already expired or the account is locked. The user can create a credential policy and associate the same with user password credentials.

After the CiscoTSP application is started, the password expiry notification appears. This information appears only once, when the application is initially loaded, and will not be updated periodically.

If the application is already started and second application begins, the password expiry notification will not appear again because the application is already loaded.

CiscoTSP notifies the user about the password expiry as follows:

Popup message from CiscoTSP notifier: **Unified CM TSP initialization Success -Password will Expire in Days : 9**

Application Event Log message: **Information : Password will Expire in [9] Days**

Password Expired

The CiscoTSP initialization fails and a message is displayed if the password is expired.

The password of a user with Credential Policy configured can expire for any of the following reasons:

- User did not reset the password before the password expiry date.
- Credential Policy was not reconfigured to increase the number of days until password expiry.

Expired Password can be reset by either the administrator or the user.

CiscoTSP notifies the user about the expired password as follows:

Popup message from CiscoTSP notifier: **Unified CM TSP initialization failed -Password has Expired, Please RESET Password**

Application Event Log message: **ERROR : Provider Open Failed as Password is expired. User can RESET the Password**

Popup message from CiscoTSP notifier: **Unified CM TSP initialization failed -Password has Expired, User cannot RESET the Password, Please contact ADMIN**

Application Event Log message: **ERROR : Provider Open Failed as Password is expired. User cannot RESET the Password, Please contact ADMIN**

Account Lock

A user account gets locked in any of the following conditions:

- Threshold number of incorrect logins is exceeded. This appears as **Failed Logon** in the user credential page.
- Administrator has locked the user account.
- Credential has not been used in a number of days as specified onthe user credential page and the account is locked due to inactivity. This appears as **Inactive Days Allowed** on the user credential page.

CiscoTSP notifies the user about the initialization failure as follows:

Popup message from CiscoTSP notifier: **Unified CM TSP initialization failed -Account is LOCKED**.

Application Event Log message: **ERROR: Provider Open Failed as User Account is LOCKED**.

Interface Changes

No interface changes.

Message Sequences

There are no message sequences.

Backward Compatibility

Feature is backward compatible.

Privacy Release

The Unified Communications Manager Privacy Release feature allows the user to dynamically alter the privacy setting. The privacy setting affects all existing and future calls on the device.Cisco Unified TSP does not support the Privacy Release feature.

Redirect to Device

This feature allows the Cisco TSP application to use the LineDevSpecific api to redirect a call to a specific device. Even if the device is sharing a phone line, only the target device rings and not the device that is sharing the phone line. This feature gets used by the PSAP Callback feature of Cisco Emergency Responder.

The new CCiscoLineDevSpecific extension - "CciscoLineDevSpecificRedirectEx" is added to support this capability for the applications. This extension has the new RedirectDeviceName field to specify the target device. This extension has the below variations of the Redirect feature:

- FAC
- CMC
- FeaturePriority
- SetUp OriginalCalled
- RedirectDirectName
- Application XML Data
- Calling Search Space

Error Description

If the redirect destination is not reachable, the LINEERR_INVALADDRESS gets returned.

Interface Change

Redirect Enhancement

Message Sequence

Redirect to Device

Redirect and Blind Transfer

The Cisco Unified TSP supports several different methods of Redirect and Blind Transfer. This section outlines the different methods as well as the differences between methods.

lineRedirect

This standard TAPI lineRedirect function redirects calls to a specified destination. The Calling Search Space and Original Called Party that Cisco Unified TSP uses for this function follows:

- Calling Search Space (CSS) — Uses CSS of the CallingParty (the party being redirected) for all cases unless the call is in a conference or a member of a two-party conference where it uses the CSS of the RedirectingParty (the party that is doing the redirect).
- Original Called Party — Not changed.

lineDevSpecific -redirect reset Original Called ID

This function redirects calls to a specified destination while resetting the Original Called Party to the party that is redirecting the call. The Calling Search Space and Original Called Party that Cisco Unified TSP uses for this function follow:

- Calling Search Space (CSS) — Uses CSS of the CallingParty (the party being redirected).
- Original Called Party — Set to the RedirectingParty (the party that is redirecting the call).

lineDevSpecific -redirect set Original Called ID

This function redirects calls to a specified destination while allowing the application to set the Original Called Party to any value. The Calling Search Space and Original Called Party that Cisco Unified TSP uses for this function follow:

- Calling Search Space (CSS) — Uses CSS of the CallingParty (the party being redirected).
- Original Called Party — Set to the preferredOriginalCalledID that the lineDevSpecific function specifies.

You can use this request to implement the Transfer to Voice Mail feature (TxToVM). Using this feature, applications can transfer the call on a line directly to the voice mailbox on another line. You can achieve TxToVM by specifying the following fields in the above request: voice mail pilot as the destination DN and the DN of the line to whose voice mail box the call is to be transferred as the preferredOriginalCalledID.

lineDevSpecific -redirect FAC CMC

This function redirects calls to a specified destination that requires either a FAC, CMC, or both. The Calling Search Space and Original Called Party that Cisco Unified TSP uses for this function follow:

- Calling Search Space (CSS) — Uses CSS of the CallingParty (the party being redirected).
- Original Called Party — Not changed.

lineBlindTransfer

Use the standard TAPI lineBlindTransfer function to blind transfer calls to a specified destination. The Calling Search Space and Original Called Party that Cisco Unified TSP uses for this function follow:

- Calling Search Space (CSS) — Uses CSS of the TransferringParty (the party that is transferring the call).
- Original Called Party — Set to the TransferringParty (the party that is transferring the call).

lineDevSpecific -blind transfer FAC CMC

This function blind transfers calls to a specified destination that requires a FAC, CMC, or both. The Calling Search Space and Original Called Party that Cisco Unified TSP uses for this function follow:

- Calling Search Space (CSS) — Uses CSS of the TransferringParty (the party that is transferring the call).

Original Called Party — Set to the TransferringParty (the party that is transferring the call).

Refer and Replaces for Phones That Are Running SIP

As part of CTI support for phones that are running SIP, TSP will support new SIP features Refer and Replaces. Refer, Refer with Replaces, Invite with Replaces represent different mechanisms to initiate different call control features. For example, Refer with Replaces in SIP terms can be translated to Transfer operation in Unified CM. Invite with Replaces can be used for Pickup call feature across SIP trunks. TSP will support event handling corresponding to the features that are initiated by Unified CM phones that are running SIP / third party phones that are running SIP. TSP will not support Refer / Replaces request initiation from the API. Because TAPI does not have Refer/Replaces feature related reason codes defined, the standard TAPI reason will be LINECALLREASON_UNKNOWN. TSP will provide new call reason to user as part of LINE_CALLINFO::dwDevSpecificData if the application negotiated extension version greater or equal to 0x00070000.

For In-dialog refer, TSP places Referrer in LINECALLSTATE_UNKNOWN | CLDSMT_CALL_WAITING_STATE call state with extended call reason as CtiCallReasonRefer. This helps present the Referrer's call leg such that applications cannot call any other call functions on this call. Any request on this call when it is in LINECALLSTATE_UNKNOWN | CLDSMT_CALL_WAITING_STATE will return an error as LINEERR_INVALCALLSTATE.

The Referrer must clear this call after the Refer request is initiated. If Referrer does not drop the call, Refer feature will clear this call if the refer is successful. LINECALLSTATE_IDLE with extended reason as CtiCallReasonRefer will get reported.

If Referrer does not drop the call and if Refer request fails (For example, Refer to target is busy), refer feature will restore the call between Referrer and Referee.

With Unified CM Phones that are running SIP, Unified CM makes all the existing call features transparent such that applications will get the existing events when the phone invokes a SIP feature whose behavior matches with the existing feature of Unified CM. For example, when Refer with Replaces is called by a phone that runs SIP (with both primary and secondary/consult call legs on same SIP line) within Unified CM cluster, all the events will get reported the same as Transfer feature.

Ringback on SIP 183 for Transfers

Cisco TAPI has been updated with how it responds to SIP 183 messages when a call is transferred over a gateway or trunk. When an established call gets transferred over a trunk and a SIP 183 is received, Cisco TAPI reports a RINGBACK state. When the call is answered, Cisco TAPI reports a call state of CONNECTED.

A new Cisco CallManager service parameter, **CTI Report Ringback on SIP 183 with SDP**, has been added to configure this feature. When this sevice parameter is set to **True**, the above behavior applies. This is the default setting.

If an application needs to use the legacy behavior, you can set the service parameter to **False**. Under this setting, if the call gets transferred over a gateway or trunk, Cisco TAPI will report a PROGRESSING state when Cisco Unified Communications Manager recieves a SIP 183, but after a connection is established, Cisco TAPI does not report the CONNECTED state.

Secure Conference

Prior to release 6.0(1), the security status of each call matched the status for the call between the phone and its immediately connected party, which is a conference bridge in the case of a conference call. No secured conference resource existed, so secure conference calls were not possible.

Release 7.0(1) supports a secured conference resource to enable secure conference. The Secure Conference feature lets the administrator configure the Conference bridge resources with either a non-secure mode or an encrypted signaling and media mode. If the bridge is configured in encrypted signaling and media mode, the Conference Bridge will register to Unified Communications Manager as a secure media resource. This enables the user to create a secure conference session. When the media streams of all participants who are involved in the conference are encrypted, the conference exists in encrypted mode. Otherwise, the conference exists in non-secure mode.

The overall conference security status depends on the least-secure call in the conference. For example, suppose parties A (secure), B (secure), and C (non-secure) are in a conference. Even though the conference bridge can support encrypted sRTP and is using that protocol to communicate with A and B, C remains a non-secure phone. Thus, the overall conference security status is non-secure. Even though the overall conference security status is non-secure, because a secure conference bridge was allocated, all secure phones (in this case, A and B) will receive sRTP keys. TSP informs each participant about the overall call security status. The system provides the overall call security level of the conference to the application in the DEVSPECIFIC portion of LINECALLINFO to indicate whether the conference call is encrypted or non-secure.

The Secure Conference feature uses four fields to present the call security status:

```
DWORD CallSecurityStatusOffset;
DWORD CallSecurityStatusSize;
DWORD CallSecurityStatusElementCount;
DWORD CallSecurityStatusElementFixedSize;
```

The offset will point to following structure:

```
typedef struct CallSecurityStatusInfo
{
    DWORD CallSecurityStatus;
} CallSecurityStatusInfo;
```

The system updates LINECALLINFO whenever the overall call security status changes during the call because a secure or non-secure party joins or leaves the conference.

A conference resource gets allocated to the conference creator based on the creator security capability. If no conference resource with the same security capability is available, the system allocates a less-secure conference resource to preserve existing functionality.

When a shared line is involved in the secure conference, the phone that has its line in RIU (remote in use) mode will not show a security status for the call. However, TSP exposes the overall security status to the application along with other call information for the inactive call. This means that TSP also reports the OverallSecurityStatus to all RIU lines. The status will match what is reported to the active line. Applications can decide whether to expose the information to the end user.

Secure RTP

The secure RTP (SRTP) feature allows Cisco TSP to report SRTP information to application as well as allow application to specify SRTP algorithm IDs during device registration. The SRTP information that Cisco TSP provides will include master key, master salt, algorithmID, isMKIPresent, and keyDerivation. To receive those key materials, administrator needs to configure TLS Enabled and SRTP Enabled flag in Unified Communications Manager Admin User windows and establish TLS link between TSP and CTIManager.

Besides, during device registration, application can provide SRTP algorithm IDs for CTI port and CTI Route Point in case of media termination by application. Application should use new Cisco extension for Line_devSpecific -CciscoLineDevSpecificUserSetSRTPAlgorithmID to set supported SRTP algorithm IDs after calling LineOpen with 0x80070000 version or higher negotiated, then followed by either CCiscoLineDevSpecificUserControlRTPStream or CciscoLineDevSpecificPortRegistrationPerCall to allow TSP to open device on CTI Manager.

When call arrives on an opened line, TSP will send LINE_CALLDEVSPECIFIC event to application with secure media indicator; then, application should query LINECALLINFO to get detail SRTP information if SRTP information is available. The SRTP information resides in the DevSpecific portion of the LINECALLINFO structure.

In case of mid-call monitoring, Cisco TSP will send LINE_CALLDEVSPECIFIC with secure media indicator, however there will be no SRTP information available for retrieval under this scenario. The event is only sent upon application request via PhoneDevSpecific with CPDST_REQUEST_RTP_SNAPSHOT_INFO message type.

To support SRTP that is using static registration, a generic mechanism for delayed device/line now exists. The following ones apply:

- Extension version bit SELSIUSTSP_LINE_EXT_VER_FOR_DELAYED_OPEN = 0x40000000
- CiscoLineDevSpecificType -SLDST_SEND_LINE_OPEN
- CCiscoLineDevSpecific -CciscoLineDevSpecificSendLineOpen

If application negotiates with 0x00070000 in lineOpen against CTI port, TSP will do LineOpen/DeviceOpen immediately. If application negotiates with 0x40070000 in LineOpen against CTI port, TSP will delay the LineOpen/DeviceOpen. Application can specify SRTP algorithm ID by using CciscoLineDevSpecificUserSetSRTPAlgorithmID (SLDST_USER_SET_SRTP_ALGORITHM_ID). However, to trigger actual device/line open in TSP, application needs to send CciscoLineDevSpecificSendLineOpen(SLDST_SEND_LINE_OPEN)

If application negotiates with 0x80070000 in LineOpen against CTI port/RP, TSP will delay the LineOpen/DeviceOpen until application specifies media information in CCiscoLineDevSpecific; however, application can use CciscoLineDevSpecificUserSetSRTPAlgorithmID (SLDST_USER_SET_SRTP_ALGORITHM_ID) to specify SRTP algorithm ID before specifying the media information.

If application negotiates with 0x40070000 in LineOpen against RP, TSP should fail CciscoLineDevSpecificUserSetSRTPAlgorithmID (SLDST_USER_SET_SRTP_ALGORITHM_ID) request because RP should have media terminated by application.

Currently, the SELSIUSTSP_LINE_EXT_VER_FOR_DELAYED_OPEN bit only gets used on CTI port when TSP Wave Driver is used to terminate media. Under conference scenario, the SRTP information gets stored in conference parent call for each party. An application that negotiates with correct version and is interested in the SRTP information in a conference scenario should retrieve the SRTP information from CONNECTED call of the particular conference party.

Backward Compatibility

CCiscoLineDevSpecific extension

CciscoLineDevSpecificUserSetSRTPAlgorithmID is defined.

CCiscoLineDevSpecific extension CciscoLineDevSpecificSendLineOpen is defined. An extra LINE_CALLDEVSPECIFIC event gets sent if negotiated version of application supports this feature while keeping existing LINE_CALLDEVSPECIFIC for reporting existing RTP parameters.

Wave driver (media terminating endpoint) uses the strip_policy to create a crypto context. It should match the encrypt and decrypt packets sent/received by IPPhones/CTIPorts. Phone uses one hardcoded srtp_policy for all phone types including phones that are using SIP.

policy->cipher_type = AES_128_ICM;

policy->cipher_key_len = 30;

policy->auth_type = HMAC_SHA1;

policy->auth_key_len = 20;

policy->auth_tag_len = 4;

policy->sec_serv = sec_serv_conf_and_auth;

Note Applications should not store key material and must use the proper security method to ensure confidentiality of the key material. Applications must clear the memory after key information is processed. Be aware that applications are subject to export control when they provide mechanism to encrypt the data (SRTP). Applications should get export clearance for that.

Presentation Indication

Secure TLS

Establishing secure connection to CTIManager involves application user to configure more information through Cisco TSP UI. This information will help TSP to create its own client certificate. This certificate is used to create a mutually authenticated secure channel between TSP and CTIManager.

TSP UI adds a new tab called Security and the options that are available on this tab follows:

- Check box for Secure Connection to CTIManager: If checked, TSP will connect over TLS CTIQBE port (2749); otherwise, TSP will connect over CTIQBE port (2748).
- Default setting specifies non secure connection and the setting will remain unchecked.

Ensure that the security flag for the TSP user is enabled through Unified Communications Manager Administration as well. CTIManager will perform a verification check whether a user who is connecting on TLS is allowed to have secure access. CTIManager will allow only security enabled users to connect to TLS port 2749 and only non secure users to connect to CTIQBE port 2748.

The user flag to enable security depends on the cluster security mode. If cluster security mode is set to secure, user security settings will have a meaning; otherwise, the connection has to be non secure. If secure connection to CTIManager is checked, the following settings will get enabled for editing.

- CAPF Server: CAPF server IP address from which to fetch the client certificate.
- CAPF Port: (Default 3804) – CAPF Server Port to connect to for Certificate download.
- Authorization Code (AuthCode): Required for Client authentication with CAPF Server and Private Key storage on client machine.
- Instance ID(IID): Each secure connection to CTIManager must have its own certificate for authentication. With the restriction of having a distinct certificate per connection, CAPF Server needs to verify that the user with appropriate AuthCode and IID is requesting the certificate. CAPF server will use AuthCode and IID to verify the user identity. After CAPF server provides a certificate, it clears the AuthCode to make sure only one instance of an app requests a certificate based on a single AuthCode. CCM admin will allow user configuration to provide multiple IID and AuthCode.
- TFTP Server: TFTP server IP address to fetch the CTL file. CTL, which file is required to verify the server certificate, gets sent while mutually authenticating the TLS connection.
- Check box to Fetch Certificate: This setting is not stored anywhere, instead only gets used to update the Client certificate when it is checked and will get cleared automatically.
- Number of Retries for Certificate Fetch: This indicates the number of retries TSP will perform to connect to CAPF Server for certificate download in case an error. (Default -0) (Range – 0 to 3)
- Retry Interval for Certificate Fetch: This will be used when the retry is configured. It indicates the (secs) for which TSP will wait during retries. (Default – 0) (Range – 0 to 15)

Because user is not expected to update the client certificate every time it changes, TSP UI will pop up a message when this box is checked by user that says “This will trigger a certificate update. Please make sure that you really want to update the TSP certificate now.” This will ensure that if user selects this check box in

an error. TSP will fail to establish a secure connection to CTIManager if a valid certificate cannot be obtained. Each secure connection to CTIManager needs to have a unique certificate for authentication.

If an application tries to create more than one Provider simultaneously with the same certificate or when a session with the same certificate already exists/is open, CTI Manager disconnects both providers. TSP will try reconnecting to CTIManager to bring the provider in service. However, if both providers continuously try to connect by using the same duplicated certificate, both providers will be closed after a certain number of retries, and the certificate will be marked as compromised by CTIManager on Unified CM server. The number of retries after which the certificate should be marked as compromised is configurable from the CTIManager Service Parameter CTI InstanceId Retry Count. CTI manager rejects further attempt to open provider with the certificate that is compromised. In this case, the CAPF profile of the compromised certificate should be deleted and a new CAPF Profile must be created for the user. The new CAPF profile for the user should use new instance ID. Otherwise, the old certificate, which was compromised previously, can be used again.

A new error code, TSPERR_INIT_CERTIFICATE_COMPROMISED, with value as 0x00000011 where TSPERR_UNKNOWN is 0x00000010 now exists. Application should not have checks like “if (err < TSPERR_UNKNOWN))” because error codes exists that have a value greater than that.

When TLS is used, the initial handshake will be slow as expected due to heavy use of public key cryptography. After the initial handshake is complete and the session is established, the overhead gets significantly reduced. The following profiling result applies on ProviderOpen for both secure and non-secure CTI connection.

Controlled Devices	Type of CTI Connection	Duration on ProviderOpen	Duration on OpenAllLines	Comments
0	Non-Secure	1 sec 382 ms	N/A	
	Secure	4 sec 987 ms	N/A	With certificate retrieval.
	Secure	3 sec 736 ms	N/A	
100	Non-secure	1 sec 672 ms	3 sec 164ms	
	Secure	5 sec 758 ms	3 sec 445ms	
2500	Non-Secure	29 sec 513 ms	3 min 26 sec 728 ms	
	Secure	34 sec 219 ms	3 min 26 sec 928 ms	

Support for RSHA12 Algorithm

Prior to the11.5 release, Cisco TSP had hardcoded the digest algorithm used for validating the signature of CTLFile to SHA1. From 11.5 release onwards, Cisco TSP is enhanced to use the digest algorithm based on DigestAlgorithm Tag mentioned in CTLFile. This way Cisco TSP validates CTLFile signed with SHA1 or SHA512.

Backward Compatibility Issues:

Cisco TSP versions prior to 11.5 will not be able to validate CTLFile signed using SHA512 if the parameter: “TFTP File Signature Algorithm Required Field” (System-> Enterprise Parameters Configuration) in Cisco Unified Communications Manager is set to RSASHA512. The secure connection fails as CiscoTSP supports only SHA1 and not SHA512 algorithm.

Secured Monitoring and Recording

This feature enhances the ability to monitor or record calls in a secure environment. In Unified Communications Manager (Cisco Unified CM), Release 8.0(1), Cisco Unified CM software has been enhanced to enable call monitoring and recording in a secure environment. So, secured calls can be monitored and recorded.

The recording and monitoring session is created in a secure mode only when the Supervisor/Recording Device and the Agent has secure capabilities. Recording/Monitoring request is successful only when the Supervisor/Recording Device has higher than or meets the security capabilities of the Agent.

TransactionID, which is unique for reach monitoring session, is exposed to the application in Call Attribute information, DevSpecific part of Call Info for the call on supervisor and agent.

If the Silent Monitoring session is toned down when the Supervisor Security capabilities do not meet or exceed the capabilities of the agent, LINE_DEVSPECIFIC event is delivered with Param1 = SLDSMT_MONITORING_TERMINATED indicating the Monitoring Terminated event and Param2 = TransactionID of the call that is terminated.

To receive the Monitoring Terminated event, the DEVSPECIFIC_SILENT_MONITORING_TERMINATED message flag must be set in applications by using the lineDevSpecific SLDST_SET_STATUS_MESSAGES request.

The application has to determine the monitoring session to be terminated based on the TransactionID that TSP exposes in LINE_DEVSPECIFIC Event for the Monitoring Session Terminated Event:

- Monitoring Terminated event is delivered to the original supervisor that initiated the Monitoring session and is longer present in monitoring the call.
- Recording: Cisco Unified CM does not support recording on Authenticated devices and also when the Recording device is authenticated.
- If the application monitors the Customer, Agent and Supervisor lines after Monitoring/Recording sessions start, CallReason will be LINECALLREASON_UNKNOWN for direct calls from the customer to the agent. CallReason will be LINECALLREASON_DIRECT for the monitored call on Supervisor as CTI reports the CallReason = CtiReasonSilentMonitoring/ CtiReasonRecording for respective Feature.

Note CallAttribute information in devspecific part of callInfo for a call on Supervisor is not cleared when the agent drops the call, in case the Monitored call is being conferenced by two Supervisors.

Interface Changes

New error Code – LINEERR_SECURITY_CAPABILITIES_MISMATCH 0xC000000E

Existing Cause Code – LINEDISCONNECTMODE_INCOMPATIBLE 0x00000400

New LINE_DEVSPECIFIC message Type -SLDSMT_MONITORING_TERMINATED. For more information, see Silent Monitoring Session Terminated Event.

New LineDevSpecificSetStatusMessage Flag -DEVSPECIFIC_SILENT_MONITORING_TERMINATED. For more information, see Set Status Messages.

In the CallAttributeInfo_ExtA0 field, LINECALLINFO::DEVSPECIFIC added the TransactionID field. For more information, see Details.

Message Sequences

See Secure Monitoring and Recording.

Select Calls

The Select softkey on IP phones lets a user select call instances to perform feature activation, such as transfer or conference, on those calls. The action of selecting a call on a line locks that call, so it cannot be selected by any of the shared line appearances. Pressing the “Select” key on a selected call will deselect the call.

Cisco Unified TSP does not support the “Select” function to select calls because all transfer and conference functions contain parameters that indicate on which calls the operation should be invoked.

Cisco Unified TSP supports the events that a user who selects a call on an application-monitored line causes. When a call on a line is selected, all other lines that share the same line appearance will see the call state change to dwCallState = CONNECTED and dwCallStateMode = INACTIVE.

Conference Changes

Transfer Changes

Set the Original Called Party Upon Redirect

Two extensions enable setting the original called party upon redirect as follows:

- CCiscoLineDevSpecificRedirectResetOrigCalled
- CCiscoLineDevSpecificRedirectSetOrigCalled

See lineDevSpecific for more information.

Shared Line Appearance

Cisco Unified TSP supports opening two different lines that each share the same DN. Each of these lines represents a Shared Line Appearance.

The Unified Communications Manager allows multiple active calls to exist concurrently on each of the different devices that share the same line appearance. The system still limits each device to, at most, one active call and multiple hold or incoming calls at any given time. Applications that use the Cisco Unified TSP to monitor and control shared line appearances can support this functionality.

If a call is active on a line that is a shared line appearance with another line, the call appears on the other line with the dwCallState = CONNECTED and the dwCallStateMode = INACTIVE. Even though the call party information may not display on the actual IP phone for the call at the other line, Cisco Unified TSP still reports the call party information on the call at the other line. This gives the application the ability to decide whether

to block this information. Also, the system does not allow call control functions on a call that is in the CONNECTED INACTIVE call state.

Cisco Unified TSP does not support shared lines on CTI Route Point devices.

In the scenario where a line is calling a DN that contains multiple shared lines, the dwCalledIDName in the LINECALLINFO structure for the line with the outbound call may be empty or set randomly to the name of one of the shared DNs. The reason for this should be obvious as Cisco Unified TSP and the Unified Communications Manager cannot resolve which of the shared DN's you are calling until the call is answered.

Silent Install

The Cisco TSP installer supports silent install, silent upgrade, and silent reinstall from the command prompt. Users do not see any dialog boxes during the silent installation.

Silent Monitoring

Silent monitoring is a feature that enables a supervisor to listen to a conversation between an agent and a customer without the agent detecting the monitoring session. TSP provides monitoring type in line DevSpecific request for applications to monitor calls on a per call basis. Both monitored and monitoring party have to be in controlled list of the user.

The Application is required to send permanent lineID, monitoring Mode and toneDirection as input to CCiscoLineDevSpecificStartCallMonitoring. Only silent monitoring mode is supported and the supervisor cannot talk to the agent.

The Application can specify if a tone should be played when monitoring starts. ToneDirection can be none (no tone played), local (tone played to Agent only), remote (tone played to Customer and Supervisor), both local and remote (tone played to agent, customer, and supervisor).

```
enum PlayToneDirection
{
 PlayToneDirection_LocalOnly = 0,
 PlayToneDirection_RemoteOnly,
 PlayToneDirection_BothLocalAndRemote,
 PlayToneDirection_NoLocalOrRemote
};
```

Monitoring of a call which is in connected state on that line will start if the request is successful. This will result in a new call between supervisor and agent. However, the call will be automatically answered with Built-in Bridge (BIB) and agent is not be aware of the call. The call established between supervisor and agent will be one-way audio call. Supervisor will get the mixed stream of agent and customer voices. The application can only invoke the monitoring session for a call after it becomes active.

TSP will send LINE_CALLDEVSPECIFIC (SLDSMT_MONITORING_STARTED) event to the agent call when supervisor starts monitoring the call. TSP will provide monitored party's call attribute information (deviceName, DN, Partition) to the monitoring party in DEVSPECIFIC portion of LINECALLINFO after monitoring has started. Similarly, TSP will provide monitoring party's call attribute information (deviceName, DN, Partition) to the monitored party in devspecific data of LINECALLINFO after monitoring has started.

The monitoring session will be terminated when the agent-customer call is ended by either the agent or the customer. The supervisor can also terminate the monitoring session by dropping the monitoring call.TSP will

inform agent by sending LINE_CALLDEVSPECIFIC (SLDSMT_MONITORING_ENDED) when supervisor drops the call. The event will not be sent if monitoring session has been ended after agent dropped the call.

SIP URL Address

As part of CTI support for phones that are using SIP, TSP will expose SIP URL that is received in Device/Call event that is received from CTIManager. The SIP URL will get presented for each corresponding party in extended call information structure of LINE_CALLINFO::dwDevSpecificData.

When a SIP trunk is involved in a call, the DN may not get presented in party information. Application then needs to consider SIP URL information under this call scenario for information.

TSP will provide SIP URL information to user as part of LINE_CALLINFO::dwDevSpecificData if the application negotiated extension version greater than or equal to 0x00070000.

CTI phones that are running SIP support the following features or functions:

- Call Initiate
- Call Answer
- Call Close/Disconnect
- Consult Transfer
- Direct Transfer
- Call Join
- Conference
- Hold/unhold
- Line Dial
- Redirect
- lineDevSpecific (SLDST_MSG_WAITING)
- lineDevSpecific (SLDST_MSG_WAITING_DIRN)

Presentation Indication

Change Notification of SuperProvider and CallPark DN Monitoring Flags

Super Provider

The Super Provider functionality allows a TSP application to control any CTI controllable device in the system (IP Phones, CTI Ports, CTI Route Points etc). The TSP application has to have an associated list of devices that it can control. It cannot control any devices that are outside of this list. However, certain applications would want to control a large number (possibly all) of the CTI controllable devices in the system. Super Provider enables the administrator to configure a CTI application as a "Super-Provider." This will mean that particular application can control absolutely any CTI controllable device in the system.

Previously, the Super Provider functionality was never exposed to TSP apps, that is the TSP application needed to have the device in the controllable list. In this release, TSP apps can control any CTI controllable device in the CallManager system. The Super-Provider apps need to explicitly "Acquire" the device before opening them.

TSP exposes new interfaces to the apps to explicitly Acquire/Deacquire any device in the system. The device information will be fetched for the explicitly acquired device using the SingleGetInfoFetch API exposed via QBE by CTI. Later, TSP will fetch the line information for this device using the LineInfoFetch API. The lines of this device are reported to the app using the LINE_CREATE/PHONE_CREATE events.

The apps can explicitly 'De-Acquire' the device. After the device is successfully de-acquired, TSP will fire LINE_REMOVE/PHONE_REMOVE events to the apps.

TSP also supports Change Notification of Super-Provider flag. If the administrator has configured a User as a Super-Provider in the admin pages, then the status of this is changed and the user is no more a Super-Provider, then CTI will inform the same to TSP in ProviderUserChangedEvent.

If any device had been explicitly acquired and opened in super-provider mode, then TSP will fire PHONE_REMOVE/LINE_REMOVE to the app indicating that the device/line is no more available for the app to use.

In this release, TSP supports change notification of CallParkDN Monitoring as well. Say, the user has been configured to allow monitoring of CallParkDN in the admin pages, now the status of this flag is disabled. Then TSP will fire LINE_REMOVE for the CallParkDNs.

Say, initially the CallParkDN Monitoring is disabled, now the status is changed to enabled, then TSP will fetch all the CallParkDNs from CTI and fire LINE_CREATE for each of the CallParkDNs.

SuperProvider

SuperProvider functionality allows a TSP application to control any CTI-controllable device in the system (IP Phones, CTI Ports, CTI Route Points and so on). Normally, a TSP application must have an associated

list of devices that it can control. It cannot control any devices that are outside this list; however, certain applications would want to control a large number (possibly all) the CTI controllable devices in the system.

SuperProvider functionality enables the administrator to configure a CTI application as a SuperProvider. This will mean that particular application can control absolutely any CTI controllable device in the system.

The SuperProvider functionality never gets exposed to TSP apps; that is, TSP application needed to have the device in the controllable list. In release 5.1 and later, TSP apps can control any CTI-controllable device in the Unified CM system.

The SuperProvider apps need to explicitly acquire the device before opening them. TSP exposes new interfaces to the apps to explicitly Acquire/Deacquire any device in the system. The device information is fetched for the explicitly acquired device by using the SingleGetInfoFetch API exposed via QBE by CTI. Later, TSP will fetch the line information for this device by using the LineInfoFetch API. The lines of this device are reported to the app by using the LINE_CREATE/PHONE_CREATE events.

The apps can explicitly 'De-Acquire' the device. After the device is successfully de-acquired, TSP will fire LINE_REMOVE/PHONE_REMOVE events to the apps.

TSP also supports Change Notification of "Super-Provider" flag. If the administrator has configured a User as a "Super-Provider" in the Unified CM Administration, the status of this changes and the user no longer represents a Super-Provider, then CTI will inform TSP in ProviderUserChangedEvent. If any device had been explicitly acquired and opened in super-provider mode, TSP will fire PHONE_REMOVE/LINE_REMOVE to the app and indicates that the device/line is no longer available for the app to use.

In release 5.1 and later, TSP supports change notification of CallParkDN Monitoring as well. If the user has been configured to allow monitoring of CallParkDN in the Unified CM Administration, the status of this flag is disabled. Then TSP will fire LINE_REMOVE for the CallParkDNs.

If the CallParkDN Monitoring is disabled, the status changes to enabled, TSP fetches all the CallParkDNs from CTI and fire LINE_CREATE for each of the CallParkDNs.

Support for Cisco Unified IP Phone 6900 and 9900 Series

In this release, CiscoTSP exposes Max Calls, Busy Trigger / Line Label, Line Instance, and Voice Mail Pilot Number in LineDevCap::DevSpecific interface.

TSP handles new device information -Device IP Address (IPv4 and IPv6) and NewCallRollOver/Consult call rollover/Join/DT/JAL/DTAL flag. This device information is kept in Device object and exposed through PhoneDevCap::DevSpecific, and also be exposed to Line App through LineDevCap::DevSpecific.

For NewCallRollOver/Consult call rollover/Join/DT/JAL/DTAL flag, there are two sets of information representing device setting and application behavior.

TAPI reports any change in the information above through LineDevSpecific event or PhoneDevSpecific event:

- Max Calls

  TAPI exposes Max Calls information in MaxCalls field of LineDevCaps::DevSpecific

  When the information changes, TSP reports the LineDevSpecific event with param1 = SLDSMT_LINE_PROPERTY_CHANGED, param2 has LPCT_MAX_CALLS bit on.

- Busy Trigger

  TAPI exposes busy trigger information in BusyTrigger field of LineDevCaps::DevSpecific

When the information changes, TSP reports LineDevSpecific event with param1 = SLDSMT_LINE_PROPERTY_CHANGED, param2 has LPCT_BUSY_TRIGGER bit on.

- Line Instance ID

  TAPI exposes Line Instance ID (Line Button number) of the line configured on the device in LineInstanceNumber of LineDevCaps::DevSpecific

  When the information changes, TSP reports LineDevSpecific event with param1 = SLDSMT_LINE_PROPERTY_CHANGED, param2 has LPCT_LINE_INSTANCE bit on.

- Line Label

  TAPI exposes Label of the line in LineLabelASCII and LineLabelUnicode field of LineDevCaps::DevSpecific

When the information changes, TSP reports LineDevSpecific event with param1 = SLDSMT_LINE_PROPERTY_CHANGED, param2 has LPCT_LINE_LABEL bit on.

- Voice Mail Pilot

  TAPI exposes Voice Mail Box Pilot configured on the line in VoiceMailPilotDN field of LineDevCaps::DevSpecific

  When the information changes, TSP reports LineDevSpecific event with param1 = SLDSMT_LINE_PROPERTY_CHANGED, param2 has LPCT_VOICEMAIL_PILOT bit on.

- Registered Device IP address and IP address mode

  TAPI exposes registered IP Address (IPv4 and IPv6) of the device in RegisteredIPv4Address and RegisteredIPv6Address fields of PhoneDevCaps::DevSpecific interface as well as in RegisteredIPv4Address and RegisteredIPv6Address fields of LineDevCaps::DevSpecific interface. Along with registered IP address, RegisteredIPAddressMode interface indicates whether the device is registered with IPv4 or IPv6 or IPv4 and IPv6. If the device is unregistered, the RegisteredIPAddressMode has a value of IPAddress_Unknown. In case of IPAddress_Unknown, the RegisteredIPv4Address and RegisteredIPv6Address can be used only for reference.

  Device IP address applies only to IP phones and CTI Port and RP are not supported. When the information is changed, TSP reports LineDevSpecific event with param1 = SLDSMT_LINE_PROPERTY_CHANGED, param2 has LPCT_DEVICE_IPADDRESS bit on. For phone application, TSP reports PhoneDevSpecific event with param1 = CPDSMT_PHONE_PROPERTY_CHANGED_EVENT, param2 has PPCT_DEVICE_IPADDRESS bit on

- New Call Rollover

  TAPI exposes the new call rollover information configured on the device in NewCallRollOverEnabled flag of PhoneDevCaps::DevSpecific interface as well as in NewCallRollOverEnabled flag of LineDevCaps::DevSpecific interface. There are two sets of flags, one for device and one for application.

  When the information is changed, TSP reports LineDevSpecific event with param1 = SLDSMT_LINE_PROPERTY_CHANGED, param2 has LPCT_NEWCALL_ROLLOVER bit on. For phone application, TSP reports PhoneDevSpecific event with param1 = CPDSMT_PHONE_PROPERTY_CHANGED_EVENT, param2 has PPCT_NEWCALL_ROLLOVER bit on.

- Consult Call Rollover

TAPI exposes new call rollover information configured on the device in ConsultCallRollOverEnabled flag of PhoneDevCaps::DevSpecific interface as well as in ConsultCallRollOverEnabled flag of LineDevCaps::DevSpecific interface. There are two sets of flags, one for device and one for application.

When the information changes, TSP reports LineDevSpecific event with par am1 = SLDSMT_LINE_PROPERTY_CHANGED, param2 has LPCT_CONSULTCALL_ROLLOVER bit on. For phone application, TSP reports PhoneDevSpecific event with param1 = CPDSMT_PHONE_PROPERTY_CHANGED_EVENT, param2 has PPCT_CONSULTCALL_ROLLOVER bit on.

- Join On Same Line

  TAPI exposes Join On Same Line information configured on the device in JoinOnSameLineEnabled flag of PhoneDevCaps::DevSpecific interface as well as in JoinOnSameLineEnabled flag of LineDevCaps::DevSpecific interface. There are two sets of flags, one for device and one for application.

  When changes, TSP reports LineDevSpecific event with param1 = SLDSMT_LINE_PROPERTY_CHANGED, param2 has LPCT_JOIN_ON_SAME_LINE bit on. For phone application, TSP reports PhoneDevSpecific event with param1 = CPDSMT_ PHONE_PROPERTY_CHANGED_EVENT, param2 has PPCT_JOIN_ON_SAME_LINE bit on.

- Join Across Line

  TAPI exposes Join Across Line information configured on the device in JoinAcrossLineEnabled flag of PhoneDevCaps::DevSpecific interface as well as in JoinAcrossLineEnabled flag of LineDevCaps::DevSpecific interface. There are two set of flags, one for device and one for application.

  When the information changes, TSP reports LineDevSpecific event with param1 = SLDSMT_LINE_PROPERTY_CHANGED, param2 has LPCT_JOIN_ACROSS_LINE bit on. For phone application, TSP reports PhoneDevSpecific event with param1 = CPDSMT_ PHONE_PROPERTY_CHANGED_EVENT, param2 has PPCT_JOIN_ACROSS_LINE bit on.

- Direct Transfer On Same Line

  TAPI exposes Direct Transfer On Same Line information configured on the device in DirectTransferSameLineEnabled flag of PhoneDevCaps::DevSpecific interface as well as in DirectTransferSameLineEnabled flag of LineDevCaps::DevSpecific interface. There are two sets of flags, one for device and one for application.

  When the information changes, TSP reports LineDevSpecific event with param1 = SLDSMT_LINE_PROPERTY_CHANGED, param2 has LPCT_DIRECTTRANSFER_ON_SAME_LINE bit on. For phone application, TSP reports PhoneDevSpecific event with param1 = CPDSMT_PHONE_PROPERTY_CHANGED_EVENT, param2 has PPCT_DIRECTRANSFER_ON_SAME_LINE bit on.

- Direct Transfer Across Line

  TAPI exposes Direct Transfer Across Line information configured on the device in DirectTransferAcrossLineEnabled flag of PhoneDevCaps::DevSpecific interface as well as in DirectTransferAcrossLineEnabled flag of LineDevCaps::DevSpecific interface. There are two set of flags, one for device and one for application.

  When the information is changed, TSP reports LineDevSpecific event with param1 = SLDSMT_LINE_PROPERTY_CHANGED, param2 has LPCT_DIRECTTRANSFER_ACROSS_LINE bit on. For phone application, TSP reports PhoneDevSpecific event with param1 = CPDSMT_PHONE_PROPERTY_CHANGED_EVENT, param2 has PPCT_DIRECTRANSFER_ACROSS_LINE bit on. To receive the

PHONE_PROPERTY_CHANGED_EVENT, the application must use CCiscoPhoneDevSpecificSetStatusMsgs to set the DEVSPECIFIC_DEVICE_PROPERTY_CHANGED_EVENT bit.

For the change notification event described above, the event is delivered to the application by TAPI layer only if the line or device is opened (even though CisoTSP sends the event to TAPI layer). If the line or phone is not opened, the application should call LineGetDevCaps again to obtain latest information about the line/device. The new extension 0x00090001 must be opened or negotiated for this feature.

Interface Changes

See LINEDEVCAPS, Line Property Changed Events, and Set Status Msgs.

Message Sequences

See Support for Cisco Unified IP Phone 6900 and 9900 Series Use Cases.

Backward Compatibility

This feature is backward compatible. To maintain backward compatibility new extension (0x00090001) is added.

Support for 100 + Directory Numbers

This feature enables users to have more than 100 Directory Numbers associated with a Device (Phones, CTI Ports and Route Points). TSP supports this feature and displays the corresponding Lines to the application.

Interface Changes

There are no interface changes.

Message Sequences

There are no message sequences.

Backward Compatibility

This feature is backward compatible.

Swap and Cancel Softkeys

The following softkeys have been added to the Cisco Unified IP Phone 7900 Series:

- Swap
- Cancel

Swap

The Swap softkey can be only be used when you use the Transfer or Conference feature. When you press Swap, the phone puts the consultative call on hold and resumes the primary call. Swap operation breaks the

internal linkage between the primary and consultative calls, but you can still complete the transfer or conference operation.

Cancel

When you press Cancel before completing the transfer operation, the TSP receives an event notification from CTI and cancels any pending transfer or conference requests.

The Swap and Cancel features operate as follows:

- For swap operation, the primary call state is changed to CONNECTED, and the consult call state is changed to ONHOLD.
- For cancel operation, the primary call state is changed to ONHOLD, and consult call state remains as CONNECTED.
- To complete the transfer operation after swap or cancel, the application invokes LineCompleteTransfer or CciscoLineDevSpecificDirectTransfer.
- To complete the conference operation after swap or cancel, the application invokes Cisco Join API – CCiscoLineDevSpecificJoin.

When using the Swap and Cancel features, the Cisco Unified IP Phones maintain a consulting relationship between the primary and the consulting calls, on invoking consult transfer or consult conference:

- The Swap operation puts the active call on hold and retrieves the held call.
- The Cancel operation breaks the consulting relationship between the primary and the consulting calls.

When users perform the swap operation, the behavior remains the same while resuming calls and all pending transfer or conference operation are cancelled. Users can swap or toggle during consultative transfer or conference transactions, and also swap or toggle between call sessions during the transaction to check the status of each party.

Interface Changes

There are no interface changes for this feature.

Message Sequences

See Refer and Replace Scenarios.

Backward Compatibility

This feature is backward compatible.

Translation Pattern

Warning TSP does not support the translation pattern because it may cause a dangling call in a conference scenario. The application needs to clear the call to remove this dangling call or simply close and reopen the line.

Presentation Indication

Change Notification of SuperProvider and CallPark DN Monitoring Flags

Unicode

Cisco TSP supports unicode character sets. TSP will send unicode party names to the application in all call events. The party name needs to be configured in Unified Communications Manager Administration. This support is limited to only party names. The locale information also gets sent to the application. The UCS-2 encoding for unicode gets used.

The party names will exist in the DevSpecific portion of the LINECALLINFO structure. In SIP call scenarios, where a call comes back into Unified CM from SIP proxy over a SIP trunk, only ASCII name will get passed because SIP has no way of populating both ASCII and unicode. As the result, the Connected and Redirection Unicode Name will get reported as empty to application.

Unrestricted Unified CM

Encryption is permanently disabled in Unrestricted Unified Communications Manager. In the current Unified Communications Manager, signaling and media encryption are configurable. Upgrading from the Unrestricted version to the Restricted version of Unified Communications Manager is not supported. Administrator cannot create a new role with security groups and roles ("Standard CTI Secure Connection" and "Standard CTI Allow Reception of SRTP Key Material") as these roles are not available in Unrestricted Unified Communications Manager.

Note Upgrade from unrestricted version to the restricted version is not supported.

In case of an upgrade from Restricted to Unrestricted Unified Communications Manager, all the security features are disabled and Standard CTI Secure roles associated with the end user are removed. But the custom administrative user roles created with the CTI secure privileges mentioned above are not disabled in the Unified Communications Manager database. In such cases, the application can connect to Unrestricted Unified Communications Manager as a non-secure application because CTIManager filters the information about CTI secure roles.

Also, after an upgrade from Restricted Unified Communications Manager to Unrestricted Unified Communications Manager, secure TAPI application cannot connect to Unrestricted Unified Communications Manager. To connect to the Unrestricted Cisco Unified Communications Manager after an upgrade, the application must disable secure connection from TSP UI.

If the application tries to register CTI Ports/Route Points as secure endpoints in an Unrestricted Unified Communications Manager, then the request fails. However, in some scenarios the registration request may pass, but the device remains closed and failure is reported to the application.

Interface Changes

There are no interface changes for this feature.

Message Sequences

See Unrestricted Unified CM.

Backward Compatibility

This feature is backward compatible.

URI Dialing

Cisco TSP supports dialing using directory URIs as the destination address. Cisco TSP uses the @ symbol to differentiate between directory URIs and directory numbers. If an @ symbol is present, the dialed address is a directory URI. Directory URIs can now be returned in the dwDevSpecificData call structure.

URI dialing is also supported for CTI Remote Devices. Remote destinations can be configured with directory URIs as the remote destination number.

Interface Changes

The following interfaces have changed to support directory URIs:

- lineMakeCall—lpszDestAddress parameter can contain a directory URI
- lineBlindTransfer—lpszDestAddress parameter can contain a directory URI
- lineForward—dwDestAddressOffset in the lineForward structure can now point at a destination address
- lineRedirect—lpszDestAddress parameter can contain a directory URI

Message Sequence Charts

There is no change to the message sequence.

Backwards Compatibility

No backwards compatibility issues.

Video On Hold Support

CiscoTSP is enhanced to provide capability for applications to select/Play Video file when call is placed on Hold. This enhancement was designed for the Remote Expert solution. The newly added contentID is a pass through from application (TAPI) to CCM. TAPI will not process or manipulate this value. The contentID references a VoH stream to be played when the call is placed on hold.

The VoH files are housed externally on a media sense server. To have video on hold capability, the video on hold server must be configured in CCM Admin. This server coincides to the media sense server which houses all the VoH files.

New CCiscoLineDevSpecific extension - "**CciscoLineDevSpecificHoldEx**" is added to support this capability for the applications.

Interface Changes

See lineHold Enhancement.

Use Cases

See LineHold Enhancement.

Backward Compatibility

There are no backward compatibility issues for this feature.

Whisper Coaching

Silent Call Monitoring is a feature that allows a supervisor to discreetly listen to a conversation between an agent and a customer without allowing the agent to detect the monitoring session. Whisper Coaching is an enhancement to the Silent Call Monitoring feature. Whisper Coaching allows supervisors to talk to agents during a monitoring session. This feature provides applications the ability to change the current monitoring mode of a monitoring call from Silent Monitoring to Whisper Coaching and vice versa.

If the application opens the line while the whisper coaching session is already in progress, TSP exposes either the TSPCallAttribute_WhisperCoachingCall or TSPCallAttribute_WhisperCoachingCall_Target bitmap in the CallAttributeType field of LineCallInfo::DevSpecific. This indicates whether the call is a whisper coaching call or the target of a whisper coaching call.

Support of this feature consists of the following:

- In the exiting CciscoLineDevSpecificStartCallMonitoringrequest, an m_MonitorMode parameter is enhanced to support an enumeration for Whisper coaching.
- A new CCiscoLineDevSpecific extension, CciscoLineDevSpecificMonitoringUpdateMode, needs to be added that allows the application to toggle between the silent monitoring and whisper coaching modes and vice versa.
- A new LineCallDevSpecific event updates the application stating that the monitoring mode has changed.
- A new field, activeToneDirection, is added in the CallAttributeInfo_ExtA0 structure. This field specifies the active tone direction that is being played for the call.

The behaviors of toneDirection are as follows:

- The existing service parameters for toneDirection are used for both silent and whisper coaching monitoring sessions. The service parameters are “Play Monitoring Notification Tone To Observed Target” (for a local party or an agent) and “Play Monitoring Notification Tone To Observed Connected Parties” (for a remote party or a customer).
- For CciscoLineDevSpecificStartCallMonitoring (monitoring mode = Silent) or CciscoLineDevSpecificMonitoringUpdateMode (monitoring mode = Silent), the application specified toneDirection is used in addition to the toneDirection configured with the service parameters.

- For CciscoLineDevSpecificStartCallMonitoring (monitoring mode = Whisper) or CciscoLineDevSpecificMonitoringUpdateMode (monitoring mode = Whisper), the application specified toneDirection for a remote side (customer only) is honored. This feature uses the service parameter that configured play toneDirection and the application specified toneDirection to play the tone to the remote side (customer only).

The following table lists effective toneDirection for StartSilentMonitoring/Toggle to SilentMonitoring.

Table 1: Effective ToneDirection for StartSilentMonitoring/Toggle to SilentMonitoring

	Observed Target = false **Observed Connected Parties = false**	**Observed Target = true** **Observed Connected Parties = false**	**Observed Target = false** **Observed Connected Parties = true**	**Observed Target = true** **Observed Connected Parties = true**
AppRequests toneDirection = None	None	Local Only	Remote Only	Both
AppRequests toneDirection = Local Only	Local Only	Local Only	Both	Both
AppRequests toneDirection = Remote Only	Remote Only	Both	Remote Only	Both
AppRequests toneDirection = Both	Both	Both	Both	Both

The following table lists effective toneDirection for Start WhisperCoaching/Toggle to WhisperCoaching.

Table 2: Effective ToneDirection for StartWhisperCoaching/Toggle to WhisperCoaching

	Observed Target = false **Observed Connected Parties = false**	**Observed Target = true** **Observed Connected Parties = false**	**Observed Target = false** **Observed Connected Parties = true**	**Observed Target = true** **Observed Connected Parties = true**
AppRequests toneDirection = None	None	None	Remote Only	Remote Only
AppRequests toneDirection = Local Only	None	None	Remote Only	Remote Only
AppRequests toneDirection = Remote Only	Remote Only	Remote Only	Remote Only	Remote Only
AppRequests toneDirection = Both	Remote Only	Remote Only	Remote Only	Remote Only

Due to the fix for CSCsb89374, the behavior for the toggle request using CciscoLineDevSpecificMonitoringUpdateMode is different between SIP and SCCP phones. When CciscoLineDevSpecificMonitoringUpdateMode is changed:

- For SCCP phones, no media break is reported
- For SIP phones, a media break is always reported

Consider the following example for toggle:

```
dwParam1 = 0x30403 represents StartReception
dwParam1 = 0x e0500401 represents StartTransmission
dwParam1 = 0x4 represents StopReception
dwParam1 = 0x2 represents StopTransmission
```

	SIP	SCCP
Toggle from Silent to Whisper	LINE_DEVSPECIFIC dwParam1 = 0x4 LINE_DEVSPECIFIC dwParam1 = 0x30403 LINE_DEVSPECIFIC dwParam1 = 0x e0500401	LINE_DEVSPECIFIC dwParam1 = 0xe0500401
Toggle from Whisper to Silent	LINE_DEVSPECIFIC dwParam1 = 0x4 LINE_DEVSPECIFIC dwParam1 = 0x2 LINE_DEVSPECIFIC dwParam1 = 0x30403	LINE_DEVSPECIFIC dwParam1 = 0x2

Interface Changes

See UpdateMonitorMode, Monitor Mode Update Event, Details, Details, and Details.

Message Sequences

See Whisper Coaching.

Backward Compatibility

This feature is backward compatible.

XSI Object Pass Through

XSI-enabled IP phones allow applications to directly communicate with the phone and access XSI features, such as manipulate display, get user input, play tone, and so on. To allow TAPI applications access to the XSI

capabilities without having to set up and maintain an independent connection directly to the phone, TAPI provides the ability to send the device data through the CTI interface. The system exposes this feature as a Cisco Unified TSP device-specific extension.

The system only supports the PhoneDevSpecificDataPassthrough request for IP phone devices.