Résolution des problèmes EM_PARK pour signalisation CAS numérique E&M

Contenu

[Introduction](#)
[Conditions préalables](#)
[Conditions requises](#)
[Components Used](#)
[Théorie générale](#)
[Conventions](#)
[Problème](#)
[Solution](#)
[Fausse réponse](#)
[Informations connexes](#)

Introduction

Dans la signalisation E&M numérique sur les plates-formes de routeurs Cisco 2600, 3600 et MC3810, certains créneaux horaires T1/E1 peuvent être bloqués à l'état EM_PARK. Ceci est visible lorsque vous émettez la commande **show voice call summary**. Ce document explique comment résoudre ce problème.

Ce résultat montre que certains créneaux horaires sont à l'état EM_PARK. Un créneau horaire dans l'état EM_PARK n'est pas utilisé pour les appels vocaux.

```
Router#show voice call summary
PORT      CODEC     VAD    VTSP STATE     VPM STATE
=====     ======    ===    ==========     ==========
1/0:0.1     -         -          -         EM_ONHOOK
1/0:0.2     -         -          -         EM_PARK
1/0:0.3     -         -          -         EM_PARK
1/0:0.4     -         -          -         EM_ONHOOK
1/0:0.5     -         -          -         EM_ONHOOK
```

Conditions préalables

Conditions requises

Aucune spécification déterminée n'est requise pour ce document.

Components Used

Les informations contenues dans ce document sont basées sur les versions de matériel et de

logiciel suivantes :

- Matériel : routeurs Cisco 2600, Cisco 3600, Cisco VG200 et MC3810
- Logiciels - Tous

The information in this document was created from the devices in a specific lab environment. All of the devices used in this document started with a cleared (default) configuration. If your network is live, make sure that you understand the potential impact of any command.

Théorie générale

Dans le CAS T1, par exemple, la signalisation de début de liaison, lorsque le PBX est décroché, le logement de temps latéral du routeur/passerelle reste inactif (EM_ONHOOK) jusqu'à ce que l'appel soit pris en charge par une destination distante. L'état de l'emplacement temporel du routeur devient EM_OFFHOOK lorsque la destination distante répond à l'appel.

Si l'appel ne se connecte pas, le routeur/la passerelle diffuse les tonalités de réorganisation intrabande à l'appelant. Puisque l'état du canal côté routeur est toujours EM_ONHOOK, le routeur ne peut pas raccrocher le canal. Une fois l'appelant raccroché, le PBX doit changer son état de canal de décrochage à raccrochage.

Dans certains cas, les PBX n'envoient pas les messages onhook, à l'aide des transitions ABCD. Le routeur dispose d'une solution de contournement pour cette soi-disant réponse fausse. Sans la solution de contournement des fausses réponses, les canaux sont bloqués dans un état EM_PARK indéfiniment. Consultez la section Fausse réponse pour plus d'informations.

Remarque : les appels peuvent être bloqués à l'état EM_PARK sur certains canaux T1 si le châssis du routeur de la passerelle vocale n'est pas correctement mis à la terre. Reportez-vous au guide d'installation du matériel pour plus d'informations sur la mise à la terre électrique.

Conventions

Pour plus d'informations sur les conventions utilisées dans ce document, reportez-vous à Conventions relatives aux conseils techniques Cisco.

Problème

Il y a deux raisons principales possibles pour lesquelles le créneau horaire est bloqué dans l'état EM_PARK :

- Le processeur de signal numérique (DSP) est défectueux et présente des problèmes matériels ou logiciels.
- Le commutateur RTPC/PBX envoie un signal de décrochage continu au routeur et ne le libère pas.

Solution

Voici les solutions à ce problème :

Si les intervalles de temps de votre système sont bloqués à l'état EM_PARK, vérifiez les DSP.

Référez-vous à Dépannage du DSP sur NM-HDV pour les routeurs de la gamme Cisco 2600/3600 afin de vérifier les DSP.

Si les DSP sont actifs, le problème peut se trouver du côté du commutateur PSTN/PBX ou de la plate-forme logicielle Cisco IOS® (le routeur/passerelle ne lance pas la procédure de réponse erronée). Consultez la section Fausse réponse pour plus d'informations.

Fausse réponse

Le routeur/passerelle Cisco attend une valeur par défaut de 30 secondes (utilisez les commandes **timeouts wait-release** et **timeouts call-disconnect** afin de modifier ces valeurs) après avoir appris que l'intervalle de temps doit être défini du PBX sur onhook pendant qu'il lit la tonalité de réorganisation.

Si cela ne se produit pas, le routeur déplace l'intervalle de temps à l'état EM_PARK et démarre un autre minuteur d'une durée de 10 secondes. Si le PBX n'est toujours pas raccroché après la durée de 10 secondes, le routeur joue sur le PBX. Le routeur envoie une *fausse réponse* d'une seconde, puis raccroche.

Une fois que le routeur a envoyé le signal de réponse falsifié, il démarre un autre compteur de cinq minutes. Si le PBX est raccroché, le compteur s'arrête et le routeur passe l'intervalle de temps à l'état EM_ONHOOK. Sinon, après cinq minutes, il envoie un autre signal de réponse falsifié d'une durée d'une seconde. Le routeur répète ce processus jusqu'à ce que le PBX se raccroche. Le routeur force le PBX à effacer l'appel.

Note : Cette transition de réponse n'est mise à jour en aucun enregistrement comptable, car l'appel réel est effacé. Mais le PBX le comprend comme une réponse et l'utilisateur est probablement facturé pour l'appel de durée d'une seconde.

Si le DSP associé au créneau horaire dans l'état EM_PARK est actif et sain et que le problème persiste, exécutez les commandes **debug vpm all** et **debug vtsp all** afin de voir si Cisco IOS tente d'envoyer la fausse réponse.

Note : Vous devez exécuter les débogages pendant plus de cinq minutes.

Remarque : dans la plupart des cas, si le DSP est défectueux, le routeur n'effectue pas la solution de contournement des fausses réponses. Référez-vous à Dépannage du DSP sur NM-HDV pour les routeurs de la gamme Cisco 2600/3600 pour plus d'informations.

Cette sortie de débogage montre comment un créneau temporel devient bloqué dans EM_PARK et comment fonctionne la solution de contournement des fausses réponses.

```
Jan 11 17:19:00.767: htsp_dsp_message: SEND/RESP_SIG_STATUS: state=0xC timestamp
=44262 systime=31305235
Jan 11 17:19:00.767: htsp_process_event:
[4/1:1(10), EM_ONHOOK, E_DSP_SIG_1100]em_onhook_offhook htsp_setup_ind
!--- Offhook signal is received from the switch. Jan 11 17:19:00.767: [4/1:1(10)]
get_local_station_id calling num= calling name= calling time=01/11 17:19 Jan 11 17:19:00.767:
vtsp_tsp_call_setup_ind (sdb=0x62BB7B14, tdm_info=0x0, tsp_ info=0x62BB4050, calling_number=
calling_oct3 = 0x0, called_number= called_oct3 = 0x81, oct3a=0x0): peer_tag=0 Jan 11
17:19:00.767: : ev.clg.clir is 0 ev.clg.clid_transparent is 0 ev.clg.null_orig_clg is 1
ev.clg.calling_translated is false Jan 11 17:19:00.767: htsp_timer - 3000 msec Jan 11
```

17:19:00.767: vtsp_do_call_setup_ind Jan 11 17:19:00.767: vtsp_allocate_cdb,cdb 0x62DCEA70 Jan 11 17:19:00.767: vtsp_do_call_setup_ind: Call ID=112722, guid=62DC4230 Jan 11 17:19:00.767: vtsp_do_call_setup_ind: type=0, under_spec=1640890368, name=, id0=10, id1=1, id2=25038, calling=, called= subscriber=RegularLine Jan 11 17:19:00.767: vtsp_do_normal_call_setup_ind Jan 11 17:19:00.771: cc_api_call_setup_ind (vdbPtr=0x62BB7FA0, callInfo={called= ,called_oct3=0x81,calling=,calling_oct3=0x0,calling_oct3a=0x0,calling_xlated=fal se,subscriber_type_str=RegularLine,fdest=0,peer_tag=0, prog_ind=3},callID=0x62DC 40DC) Jan 11 17:19:00.771: cc_api_call_setup_ind type 1 , prot 0 Jan 11 17:19:00.771: vtsp_insert_cdb,cdb 0x62DCEA70 Jan 11 17:19:00.771: vtsp_open_voice_and_set_params Jan 11 17:19:00.771: dsp_close_voice_channel: [4/1:1:32995] packet_len=8 channel _id=3 packet_id=75 Jan 11 17:19:00.771: dsp_open_voice_channel_20: [4/1:1:32995] packet_len=16 channel_id=3 packet_id=74 alaw_ulaw_select=0 associated_signaling_channel=130 time_slot=2 serial_port=0 Jan 11 17:19:00.771: vtsp_modem_proto_from_cdb: cap_modem_proto 1073741824 Jan 11 17:19:00.771: vtsp_modem_proto_from_cdb: cap_modem_proto 1073741824 Jan 11 17:19:00.771: dsp_encap_config: [4/1:1:32995] packet_len=30 channel_id=3 packet_id=92 TransportProtocol 2 t_ssrc=0x0 r_ssrc=0x0 t_vpxcc=0x0 r_vpxcc=0x0 sid_support=1, tse_payload=65535, seq_num=0x0, redundancy=0 Jan 11 17:19:00.771: dsp_set_playout_delay Jan 11 17:19:00.771: dsp_set_playout: [4/1:1:32995] packet_len=18 channel_id=3 p acket_id=76 mode=1 initial=60 min=40 max=200 fax_nom=300 dsp_set_playout_delay_config Jan 11 17:19:00.771: dsp_set_playout_config Jan 11 17:19:00.771: mode 0, init 60, min 40, max 200 playout default Jan 11 17:19:00.771: dsp_set_playout_config:mode 0, init 60, min 40, max 200 Jan 11 17:19:00.771: dsp_set_playout_config: [4/1:1:32995] packet_len=18 channel _id=3 packet_id=76 mode=1 initial=60 min=40 max=200 fax_nom=300 Jan 11 17:19:00.771: dsp_echo_canceler_control: echo_cancel: 1 Jan 11 17:19:00.771: dsp_echo_canceler_control: [4/1:1:32995] echo_cancel 1, disable_hpf 0, flags=0x0, threshold=-21 Jan 11 17:19:00.771: dsp_echo_canceler_control: [4/1:1:32995] packet_len=12 channel_id=3 packet_id=66 flags=0x0, threshold=-21 Jan 11 17:19:00.771: set_gains: FXx/E&M: msg->message.set_codec_gains.out_gain=0 Jan 11 17:19:00.771: dsp_set_gains: [4/1:1:32995] packet_len=12 channel_id=3 packet_id=91 in_gain=0 out_gain=0 Jan 11 17:19:00.771: dsp_vad_enable: [4/1:1:32995] enable: packet_len=12 channel _id=3 packet_id=78 thresh=-38 Jan 11 17:19:00.771: cc_process_call_setup_ind (event=0x62E63ACC) Jan 11 17:19:00.771: >>>>CCAPI handed cid 32995 with tag 0 to app "DEFAULT" Jan 11 17:19:00.771: sess_appl: ev(24=CC_EV_CALL_SETUP_IND), cid(32995), disp(0) Jan 11 17:19:00.771: sess_appl: ev(SSA_EV_CALL_SETUP_IND), cid(32995), disp(0) Jan 11 17:19:00.771: ssaCallSetupInd Jan 11 17:19:00.771: ccCallSetContext (callID=0x80E3, context=0x62DFBCF0) Jan 11 17:19:00.771: ssaCallSetupInd cid(32995), st(SSA_CS_MAPPING),oldst(0), ev (24)ev->e.evCallSetupInd.nCallInfo.finalDestFlag = 0 Jan 11 17:19:00.771: ccCallSetupAck (callID=0x80E3) Jan 11 17:19:00.771: ccGenerateTone (callID=0x80E3 tone=8) Jan 11 17:19:00.771: ccCallReportDigits (callID=0x80E3, enable=0x1) Jan 11 17:19:00.771: vtsp_report_digit_control: enable=1: digit reporting enabled Jan 11 17:19:00.771: cc_api_call_report_digits_done (vdbPtr=0x62BB7FA0, callID=0x80E3, disp=0) Jan 11 17:19:00.771: : vtsp_get_digit_timeouts Jan 11 17:19:00.771: sess_appl: ev(52=CC_EV_CALL_REPORT_DIGITS_DONE), cid(32995), disp(0) Jan 11 17:19:00.771: cid(32995)st(SSA_CS_MAPPING)ev (SSA_EV_CALL_REPORT_DIGITS_DONE) oldst(SSA_CS_MAPPING)cfid(-1)csize(0)in(1)fDest(0) Jan 11 17:19:00.771: ssaReportDigitsDone cid(32995) peer list: (empty) Jan 11 17:19:00.771: ssaReportDigitsDone callid=32995 Enable succeeded Jan 11 17:19:00.771: ccGenerateTone (callID=0x80E3 tone=8) Jan 11 17:19:00.771: vtsp:[4/1:1:32995, S_SETUP_INDICATED, E_CC_SETUP_ACK] Jan 11 17:19:00.775: act_setup_ind_ack Jan 11 17:19:00.775: vtsp_modem_proto_from_cdb: cap_modem_proto 0 Jan 11 17:19:00.775: vtsp_modem_proto_from_cdb: cap_modem_proto 0 Jan 11 17:19:00.775: dsp_encap_config: [4/1:1:32995] packet_len=30 channel_id=3 packet_id=92 TransportProtocol 2 t_ssrc=0x0 r_ssrc=0x0 t_vpxcc=0x0 r_vpxcc=0x0 sid_support=1, tse_payload=65535, seq_num=0x0, redundancy=0 Jan 11 17:19:00.775: dsp_voice_mode: [4/1:1:32995] cdb 62DCEA70, cdb->codec_params.modem 2, inband_detect flags 0x21 Jan 11 17:19:00.775: map_dtmf_relay_type--digit relay mode: 2 Jan 11 17:19:00.775: dsp_voice_mode: [4/1:1:32995] packet_len=24 channel_id=3 packet_id=73 coding_type=1 voice_field_size=160 VAD_flag=0 echo_length=256 comfort _noise=1 inband_detect=33 digit_relay_mode=2 AGC_flag=0act_setup_ind_ack: modem_mode = 0, fax_relay_on = 1 Jan 11 17:19:00.775: act_setup_ind_ack(): dsp_dtmf_mode() dsp_dtmf_mode(VTSP_TONE_DTMF_MODE) Jan 11 17:19:00.775: dsp_dtmf_mode: [4/1:1:32995] packet_len=10 channel_id=3 packet_id=65 dtmf_or_mf=0 Jan 11 17:19:00.775: vtsp_timer: 31305236 Jan 11 17:19:00.775: vtsp:[4/1:1:32995, S_DIGIT_COLLECT, E_CC_GEN_TONE] Jan 11 17:19:00.775: act_gen_tone Jan 11 17:19:00.775: dsp_cp_tone_off: [4/1:1:32995] packet_len=8 channel_id=3 packet_id=71 Jan 11 17:19:00.775: dsp_cp_tone_on: [4/1:1:32995] packet_len=38 channel_id=3 packet_id=72 tone_id=4 n_freq=2 freq_of_first=350 freq_of_second=440 amp_of_first=5514 amp_of_second=5514 direction=1 on_time_first=65535 off_time_first=0 on_time_second=0 off_time_second=0 Jan 11 17:19:00.775: vtsp:[4/1:1:32995, S_DIGIT_COLLECT, E_CC_GEN_TONE] Jan 11 17:19:00.775: act_gen_tone Jan 11

17:19:00.775: dsp_cp_tone_off: [4/1:1:32995] packet_len=8 channel_id=3 packet_id=71 Jan 11 17:19:00.775: dsp_cp_tone_on: [4/1:1:32995] packet_len=38 channel_id=3 packet_id=72 tone_id=4 n_freq=2 freq_of_first=350 freq_of_second=440 amp_of_first= 5514 amp_of_second=5514 direction=1 on_time_first=65535 off_time_first=0 on_time4_second=0 off_time_second=0 Jan 11 17:19:00.775: htsp_process_event: [4/1:1(10), EM_WAIT_SETUP_ACK, E_HTSP_SETUP_ACK]em_wait_setup_ack_get_ack Jan 11 17:19:00.775: htsp_timer_stop Jan 11 17:19:00.775: htsp_timer2 - 172 msec Jan 11 17:19:00.947: htsp_process_event: [4/1:1(10), EM_WAIT_SETUP_ACK, E_HTSP_EVENT_TIMER2]em_wait_prewink_timer **Jan 11 17:19:00.947: em_offhook (0)[recEive and transMit4/1:1(10)] set signal st ate = 0x8em_onhook (200)[recEive and transMit4/1:1(10)] set signal state = 0x0**

!--- A wink of duration 200 msec is sent out to the switch. Jan 11 17:19:01.471: vtsp_process_dsp_message: MSG_TX_DTMF_DIGIT_BEGIN: digit=9, rtp_timestamp=0xED31C493 Jan 11 17:19:01.471: vtsp:[4/1:1:32995, S_DIGIT_COLLECT, E_DSP_DTMF_DIGIT_BEGIN] Jan 11 17:19:01.471: act_report_digit_begin Jan 11 17:19:01.471: cc_api_call_digit_begin (dstVdbPtr=0x0, dstCallId=0xFFFFFFF F, srcCallId=0x80E3, digit=9, digit_begin_flags=0x1, rtp_timestamp=0xED31C493 rtp_expiration=0x0, dest_mask=0x1) Jan 11 17:19:01.471: sess_appl: ev(10=CC_EV_CALL_DIGIT_BEGIN), cid(32995), disp(0) Jan 11 17:19:01.471: cid(32995)st(SSA_CS_MAPPING)ev(SSA_EV_DIGIT_BEGIN) oldst(SSA_CS_MAPPING)cfid(-1)csize(0)in(1)fDest(0) Jan 11 17:19:01.471: ssaIgnore cid(32995), st(SSA_CS_MAPPING),oldst(0), ev(10) Jan 11 17:19:01.503: vtsp_process_dsp_message: MSG_TX_DTMF_DIGIT_OFF: digit=9, duration=65 Jan 11 17:19:01.503: vtsp:[4/1:1:32995, S_DIGIT_COLLECT, E_DSP_DTMF_DIGIT] Jan 11 17:19:01.503: act_report_digit_end Jan 11 17:19:01.503: vtsp_timer_stop: 31305308 Jan 11 17:19:01.503: dsp_cp_tone_off: [4/1:1:32995] packet_len=8 channel_id=3 pa cket_id=71 Jan 11 17:19:01.503: cc_api_call_digit_end (dstVdbPtr=0x0, dstCallId=0xFFFFFFFF, srcCallId=0x80E3, digit=9,duration=65,xruleCallingTag=0,xruleCalledTag=0, dest_mask=0x1), digi t_tone_mode=0 Jan 11 17:19:01.503: htsp_digit_ready: digit = 39 Jan 11 17:19:01.503: vtsp_timer: 31305308 Jan 11 17:19:01.503: htsp_process_event: [4/1:1(10), EM_OFFHOOK, E_VTSP_DIGIT]em_offhook_digit_collect Jan 11 17:19:01.503: sess_appl: ev(9=CC_EV_CALL_DIGIT_END), cid(32995), disp(0) Jan 11 17:19:01.503: cid(32995)st(SSA_CS_MAPPING)ev(SSA_EV_CALL_DIGIT) oldst(SSA_CS_MAPPING)cfid(-1)csize(0)in(1)fDest(0) Jan 11 17:19:01.503: ssaDigit Jan 11 17:19:01.503: ssaDigit, 0. sct->digit , sct->digit len 0, usrDigit 9, digit_tone_mode=0 Jan 11 17:19:01.503: ssaDigit,1. callinfo.called , digit 9, callinfo.calling , x rulecallingtag 0, xrulecalledtag 0 Jan 11 17:19:01.503: ssaDigit, 7. callinfo.calling , sct->digit 9, result 1 Jan 11 17:19:01.603: vtsp_process_dsp_message: MSG_TX_DTMF_DIGIT_BEGIN: digit=1, rtp_timestamp=0xED31C493 Jan 11 17:19:01.603: vtsp:[4/1:1:32995, S_DIGIT_COLLECT, E_DSP_DTMF_DIGIT_BEGIN] Jan 11 17:19:01.603: act_report_digit_begin Jan 11 17:19:01.603: cc_api_call_digit_begin (dstVdbPtr=0x0, dstCallId=0xFFFFFFF F, srcCallId=0x80E3, digit=1, digit_begin_flags=0x1, rtp_timestamp=0xED31C493 rtp_expiration=0x0, dest_mask=0x1) Jan 11 17:19:01.603: sess_appl: ev(10=CC_EV_CALL_DIGIT_BEGIN), cid(32995), disp(0) Jan 11 17:19:01.603: cid(32995)st(SSA_CS_MAPPING)ev(SSA_EV_DIGIT_BEGIN) oldst(SSA_CS_MAPPING)cfid(-1)csize(0)in(1)fDest(0) Jan 11 17:19:01.603: ssaIgnore cid(32995), st(SSA_CS_MAPPING),oldst(0), ev(10) Jan 11 17:19:01.643: vtsp_process_dsp_message: MSG_TX_DTMF_DIGIT_OFF: digit=1, duration=75 Jan 11 17:19:01.643: vtsp:[4/1:1:32995, S_DIGIT_COLLECT, E_DSP_DTMF_DIGIT] Jan 11 17:19:01.643: act_report_digit_end Jan 11 17:19:01.643: vtsp_timer_stop: 31305322 Jan 11 17:19:01.643: cc_api_call_digit_end (dstVdbPtr=0x0, dstCallId=0xFFFFFFFF, srcCallId=0x80E3, digit=1,duration=75,xruleCallingTag=0,xruleCalledTag=0, dest_mask=0x1), digit_tone_mode=0 Jan 11 17:19:01.643: htsp_digit_ready: digit = 31 Jan 11 17:19:01.643: vtsp_timer: 31305322 Jan 11 17:19:01.643: htsp_process_event: [4/1:1(10), EM_OFFHOOK, E_VTSP_DIGIT]em_offhook_digit_collect Jan 11 17:19:01.643: sess_appl: ev(9=CC_EV_CALL_DIGIT_END), cid(32995), disp(0) Jan 11 17:19:01.643: cid(32995)st(SSA_CS_MAPPING)ev(SSA_EV_CALL_DIGIT) oldst(SSA_CS_MAPPING)cfid(-1)csize(0)in(1)fDest(0) Jan 11 17:19:01.643: ssaDigit Jan 11 17:19:01.643: ssaDigit, 0. sct->digit 9, sct->digit len 1, usrDigit 1, digit_tone_mode=0 Jan 11 17:19:01.643: ssaDigit,1. callinfo.called , digit 91, callinfo.calling , xrulecallingtag 0, xrulecalledtag 0 Jan 11 17:19:01.643: ssaDigit, 7. callinfo.calling , sct->digit 91, result 1 Jan 11 17:19:01.743: vtsp_process_dsp_message: MSG_TX_DTMF_DIGIT_BEGIN: digit=8, rtp_timestamp=0xED31C493 Jan 11 17:19:01.743: vtsp:[4/1:1:32995, S_DIGIT_COLLECT, E_DSP_DTMF_DIGIT_BEGIN] Jan 11 17:19:01.743: act_report_digit_begin Jan 11 17:19:01.743: cc_api_call_digit_begin (dstVdbPtr=0x0, dstCallId=0xFFFFFFF F, srcCallId=0x80E3, digit=8, digit_begin_flags=0x1, rtp_timestamp=0xED31C493 rtp_expiration=0x0, dest_mask=0x1) Jan 11 17:19:01.743: sess_appl: ev(10=CC_EV_CALL_DIGIT_BEGIN), cid(32995), disp(0) Jan 11 17:19:01.743: cid(32995)st(SSA_CS_MAPPING)ev(SSA_EV_DIGIT_BEGIN) oldst(SSA_CS_MAPPING)cfid(-1)csize(0)in(1)fDest(0) Jan 11 17:19:01.743: ssaIgnore cid(32995), st(SSA_CS_MAPPING),oldst(0), ev(10) radius_decrypt: null length Jan 11 17:19:01.843: vtsp_process_dsp_message: MSG_TX_DTMF_DIGIT_OFF: digit=8, duration=75 Jan 11 17:19:01.843: vtsp:[4/1:1:32995,

S_DIGIT_COLLECT, E_DSP_DTMF_DIGIT] Jan 11 17:19:01.843: act_report_digit_end Jan 11 17:19:01.843: vtsp_timer_stop: 31305342 Jan 11 17:19:01.843: cc_api_call_digit_end (dstVdbPtr=0x0, dstCallId=0xFFFFFFFF, srcCallId=0x80E3, digit=8,duration=75,xruleCallingTag=0,xruleCalledTag=0, dest_mask=0x1), digi t_tone_mode=0 Jan 11 17:19:01.843: htsp_digit_ready: digit = 38 Jan 11 17:19:01.843: vtsp_timer: 31305342 Jan 11 17:19:01.843: htsp_process_event: [4/1:1(10), EM_OFFHOOK, E_VTSP_DIGIT]em_offhook_digit_collect Jan 11 17:19:01.843: sess_appl: ev(9=CC_EV_CALL_DIGIT_END), cid(32995), disp(0) Jan 11 17:19:01.843: cid(32995)st(SSA_CS_MAPPING)ev(SSA_EV_CALL_DIGIT) oldst(SSA_CS_MAPPING)cfid(-1)csize(0)in(1)fDest(0) Jan 11 17:19:01.843: ssaDigit Jan 11 17:19:01.843: ssaDigit, 0. sct->digit 91, sct->digit len 2, usrDigit 8, d igit_tone_mode=0 Jan 11 17:19:01.843: ssaDigit,1. callinfo.called , digit 918, callinfo.calling , xrulecallingtag 0, xrulecalledtag 0 Jan 11 17:19:01.843: ssaDigit, 7. callinfo.calling , sct->digit 918, result -1 Jan 11 17:19:01.843: ccCallDisconnect (callID=0x80E3, cause=0x1C tag=0x0) Jan 11 17:19:01.843: vtsp:[4/1:1:32995, S_DIGIT_COLLECT, E_CC_DISCONNECT] Jan 11 17:19:01.843: act_pre_con_disconnect Jan 11 17:19:01.843: vtsp_ring_noan_timer_stop: 31305342 Jan 11 17:19:01.843: dsp_cp_tone_off: [4/1:1:32995] packet_len=8 channel_id=3 pa cket_id=71 Jan 11 17:19:01.843: dsp_voice_mode: [4/1:1:32995] cdb 62DCEA70, cdb->codec_para ms.modem 2, inband_detect flags 0x21 Jan 11 17:19:01.843: map_dtmf_relay_type--digit relay mode: 2 Jan 11 17:19:01.843: dsp_voice_mode: [4/1:1:32995] packet_len=24 channel_id=3 pa cket_id=73 coding_type=1 voice_field_size=160 VAD_flag=0 echo_length=256 comfort _noise=1 inband_detect=33 digit_relay_mode=2 AGC_flag=0 Jan 11 17:19:01.843: **dsp_cp_tone_on: [4/1:1:32995] packet_len=38 channel_id=3 pa cket_id=72 tone_id=3 n_freq=2 freq_of_first=480 freq_of_second=620amp_of_first= 5206 amp_of_second=2928 direction=1 on_time_first=250 off_time_first=250 on_time_second=0 off_time_second=0**
Jan 11 17:19:01.843: **vtsp_timer: 31305342**
Jan 11 17:19:01.843: htsp_pre_connect_disconnect, cdb = 62DCEA70 cause = 1C
!--- Since the call is disconnected because the number received is "unassigned" !--- or "invalid" the router starts to play the reorder !--- tone and a timer, which is the wait-release !--- timeout timer, starts with default 30 seconds. !--- This call is disconnected !--- prior to the connect state. Jan 11 17:19:01.843: htsp_process_event: [4/1:1(10), EM_OFFHOOK, E_HTSP_PRE_CONN_DISC] **Jan 11 17:19:31.844: vtsp_main: timer: 31308342**
!--- The wait-release timer expires after 30 seconds. **Jan 11 17:19:31.844: vtsp:[4/1:1:32995, S_WAIT_RELEASE_NC, E_TIMER]**
!--- The VTSP module is in a wait release state for that call. It also receives !--- event timer, which means that the timer expires so that it !--- goes into another state. Jan 11 17:19:31.844: act_pre_con_disc_rel htsp_release_req: cause 28, no_onhook 0 **Jan 11 17:19:31.844: htsp_process_event: [4/1:1(10), EM_OFFHOOK, E_HTSP_RELEASE_REQ]em_offhook_release**
Jan 11 17:19:31.844: htsp_timer_stop2 em_onhook (0)[recEive and transMit4/1:1(10)] set signal state = 0x0
Jan 11 17:19:31.844: htsp_timer_stop
Jan 11 17:19:31.844: em_start_timer: 400 ms
Jan 11 17:19:31.844: htsp_timer - 400 msec
!--- HTSP receives an event that requests the release of !--- the time slot and it goes into EM wait !--- onhook state. But, it cannot do anything since it says I am onhook already. !--- Also, the router starts a timer of 400 msec. **Jan 11 17:19:32.296: htsp_process_event: [4/1:1(10), EM_WAIT_ONHOOK, E_HTSP_EVENT_TIMER]em_wait_timeout**
Jan 11 17:19:32.296: em_stop_timers
Jan 11 17:19:32.296: htsp_timer_stop
Jan 11 17:19:32.296: em_start_timer: 400 ms
Jan 11 17:19:32.296: htsp_timer - 400 msec
!--- When the 400 msec timer expires, HTSP gets into EM clear pending state. !--- It also starts another timer of 400 msec. Jan 11 17:19:32.696: htsp_process_event: [4/1:1(10), EM_CLR_PENDING, E_HTSP_EVENT_TIMER]em_clr_timeout Jan 11 17:19:32.696: em_stop_timers Jan 11 17:19:32.696: htsp_timer_stop Jan 11 17:19:32.696: em_start_timer: 10000 ms Jan 11 17:19:32.696: htsp_timer - 10000 msec Jan 11 17:19:32.700: htsp_dsp_message: SEND/RESP_SIG_STATUS: state=0xC timestamp=1533 systime=31308428 Jan 11 17:19:32.700: htsp_process_event: [4/1:1(10), EM_PARK, E_DSP_SIG_1100]em_park_offhook *!--- When the 400 msec timer expires, the router puts the time slot into !--- the EM_PARK state, and it starts another timer of 10 seconds. !--- The router still sees the ABCD=1100 from the switch.* **Jan 11 17:19:42.760: htsp_process_event: [4/1:1(10), EM_PARK, E_HTSP_EVENT_TIMER]em_park_timerhtsp_report_onhook_sig**
Jan 11 17:19:42.760: em_offhook (0)[recEive and transMit4/1:1(10)] set signal st ate = 0x8em_onhook (1000)[recEive and transMit4/1:1(10)] set signal state = 0x0

```
Jan 11 17:19:42.760: htsp_timer2 - 300000 msec
Jan 11 17:19:42.760: htsp_process_event: [4/1:1(10),
EM_PARK, E_HTSP_EVENT_TIMER]em_park_timerhtsp_report_onhook_sig
Jan 11 17:19:42.760: em_offhook (0)[recEive and transMit4/1:1(10)]
set signal state = 0x8em_onhook (1000)[recEive and
transMit4/1:1(10)] set signal state = 0x0
Jan 11 17:19:42.760: htsp_timer2 - 300000 msec
!--- As seen from the timestamps, when the timer expires in ten seconds, !--- the router goes
offhook for one second (1000 msec) and then onhook. !--- It also starts another timer of 300000
msec (5 minutes).
```

Informations connexes

- Assistance technique concernant la technologie vocale
- Assistance concernant les produits vocaux et de communications unifiées
- Dépannage des problèmes de téléphonie IP Cisco
- Support et documentation techniques - Cisco Systems