Troubleshooting de Troncos de Conexão de Voz

Contents

Introduction

Os troncos de conexão de voz estabelecem chamadas de voz permanentemente, voz sobre IP (VoIP), voz sobre Frame Relay (VoFR) ou voz sobre ATM (VoATM). As chamadas serão estabelecidas assim que o roteador for ligado e a configuração for concluída. Assim que as portas de voz são ativadas, as portas de voz discam automaticamente o número de telefone fictício especificado na porta de voz e fazem uma chamada para o local. As portas de voz completam a chamada para a outra extremidade através dos correspondentes peers de discagem. Uma vez estabelecida essa conexão, no que diz respeito ao roteador, a chamada de voz está em sessão e é conectada.

Prerequisites

Requirements

Não existem requisitos específicos para este documento.

Componentes Utilizados

Este documento não se restringe a versões de software e hardware específicas.

As informações neste documento foram criadas a partir de dispositivos em um ambiente de laboratório específico. All of the devices used in this document started with a cleared (default) configuration. Se sua rede l estiver ativa, certifique-se de que você entendeu o impacto potencial de qualquer comando antes de usá-lo.

Conventions

For more information on document conventions, refer to the Cisco Technical Tips Conventions.

Problema

Os problemas comuns relacionados aos troncos são transparentes para o roteador e muito difíceis de solucionar. Os problemas comuns observados com os troncos de voz se manifestam quando uma chamada é feita sobre os troncos e nada é ouvido. Esse é um dos problemas conhecidos dos troncos de conexão e é causado por muitos problemas diferentes. Outro problema são os tons de Dual Tone Multifrequency (DTMF) que não são passados corretamente, e a sinalização de Private Branch Exchange (PBX) para PBX não é transportada corretamente. Este documento apresenta soluções para esses problemas.

Quando os caminhões de voz estão ativos e ativos, os sinais se comportam de forma diferente nos caminhões de conexão. Os comandos que você normalmente emite na porta de voz para características de sinalização não são relevantes e úteis. O tronco de voz se torna um conduíte de sinalização e retransmite o sinal através do link VoIP. Quando você usa os troncos de voz, a sinalização PBX deve corresponder fim-a-fim. No que diz respeito às duas máquinas PBX, o objetivo é fazer com que a conexão do tronco de voz pareça idêntica a uma linha T1 alugada ao PBX, com roteadores completamente transparentes enquanto um link claro é estabelecido entre os dois PBXs em todo o processo.

Quando o tronco é ativado, o tronco se torna um cabo de software e o tipo de sinal é considerado um tipo de conector. O tronco não se preocupa com o tipo de sinal usado. O tronco ainda é ativado mesmo se o sinal não corresponder em ambas as extremidades. Desde que os PBXs em ambas as extremidades façam a mesma sinalização, os troncos funcionam corretamente.

Solução

A abordagem a ser seguida ao solucionar problemas de tronco de conexão é diferente da usada para chamadas comutadas. Para ver o que realmente acontece depois que os troncos são verificados, você precisa consultar a sinalização PBX. Antes de continuar a examinar a sinalização, verifique se os troncos estão ativos e se os Processadores de Sinal Digital (DSPs) processam os pacotes de voz.

Observação: você provavelmente deseja desativar a Detecção de Atividade de Voz (VAD - Voice Activity Detection) para solucionar problemas. Depois de verificar se os troncos funcionam corretamente, você precisa examinar a sinalização de telefonia para fazer troubleshooting adicional.

Se os troncos forem estabelecidos e ninguém tentar fazer uma chamada, as mensagens de keepalive do tronco serão enviadas entre as caixas remotas. Esses keepalives verificam a conectividade do tronco e carregam as informações de sinalização de ponta a ponta. Para verificar esses keepalives, emita o comando debug vpm signal. Se houver muitos troncos, a saída dos comandos **debug vpm**, você poderá limitar a saída a uma única porta se você emitir a opção de comando **debug vpm port *x, onde "x" é a porta de voz em questão.*** Esta é a saída do comando **debug vpm signal** emitido quando você olha para todas as portas:

```
21:18:12: [3/0:10(11)] send to dsp sig DCBA state 0x0
```

```
21:18:12: [3/0:0(1)] rcv from dsp SIG DCBA state 0x0
21:18:12: [3/0:12(13)] rcv from dsp SIG DCBA state 0x0
21:18:12: [3/0:20(21)] rcv from dsp SIG DCBA state 0x0
21:18:12: [3/0:12(13)] send to dsp SIG DCBA state 0x0
21:18:12: [3/0:20(21)] send to dsp SIG DCBA state 0x0
21:18:12: [3/0:0(1)] send to dsp SIG DCBA state 0x0
21:18:12: [3/0:3(4)] rcv from dsp SIG DCBA state 0x0
21:18:12: [3/0:9(10)] rcv from dsp SIG DCBA state 0x0
21:18:12: [3/0:3(4)] send to dsp SIG DCBA state 0x0
21:18:13: [3/0:9(10)] send to dsp SIG DCBA state 0x0
21:18:13: [3/0:19(20)] rcv from dsp SIG DCBA state 0x0
```

Se você limitar isso, com o comando debug vpm port *x*, as depurações serão muito mais fáceis de interpretar, como mostrado neste exemplo:

```
21:21:08: [3/0:0(1)] rcv from dsp SIG DCBA state 0x0
21:21:12: [3/0:0(1)] send to dsp SIG DCBA state 0x0
21:21:13: [3/0:0(1)] rcv from dsp SIG DCBA state 0x0
21:21:17: [3/0:0(1)] send to dsp SIG DCBA state 0x0
21:21:18: [3/0:0(1)] rcv from dsp SIG DCBA state 0x0
21:21:22: [3/0:0(1)] send to dsp SIG DCBA state 0x0
21:21:23: [3/0:0(1)] rcv from dsp SIG DCBA state 0x0
21:21:27: [3/0:0(1)] send to dsp SIG DCBA state 0x0
21:21:28: [3/0:0(1)] rcv from dsp SIG DCBA state 0x0
21:21:32: [3/0:0(1)] send to dsp SIG DCBA state 0x0
```

Os keepalives são enviados e recebidos a cada cinco segundos. Os termos "enviado para dsp" e "recebido do dsp" são do ponto de vista do Cisco IOS®. Substitua o PBX para DSP para torná-lo mais compreensível. Essas são as mensagens que são vistas enquanto não há nenhuma atividade nos troncos. As mensagens de manutenção de atividade permitem que os roteadores em cada extremidade do circuito percebam que os troncos ainda estão ativos. Quando cinco dessas mensagens são perdidas em uma linha, o tronco cai. Uma das causas é se os troncos oscilam constantemente em uma rede. Para verificar se os keepalives de tronco de voz são enviados e recebidos, emita o comando **debug vpm trunk-sc**. Essa depuração não gera nenhuma saída até que os keepalives de tronco sejam perdidos. Este é um exemplo da saída do comando **debug vpm trunk-sc** quando keepalives são perdidos:

```
22:22:38: 3/0:22(23): lost Keepalive
22:22:38: 3/0:22(23): TRUNK_SC state : TRUNK_SC_CONN_WO_CLASS, event TRUNK_RTC_LOST_KEEPALIVE
22:22:38: 3/0:22(23): trunk_rtc_set_AIS on
22:22:38: 3/0:22(23): trunk_rtc_gen_pattern : SIG pattern 0x0
22:22:38: 3/0:22(23): TRUNK_SC, TRUNK_SC_CONN_WO_CLASS ==> TRUNK_SC_CONN_DEFAULT_IDLE
22:22:39: 3/0:13(14): lost Keepalive
22:22:39: 3/0:13(14): TRUNK_SC state : TRUNK_SC_CONN_WO_CLASS, event TRUNK_RTC_LOST_KEEPALIVE
22:22:39: 3/0:13(14): trunk_rtc_set_AIS on
22:22:39: 3/0:13(14): trunk_rtc_gen_pattern : SIG pattern 0x0
22:22:39: 3/0:13(14): TRUNK_SC, TRUNK_SC_CONN_WO_CLASS ==> TRUNK_SC_CONN_DEFAULT_IDLE
```

Se nenhuma saída for vista quando o comando debug vpm trunk-sc for emitido, nenhuma manutenção de atividade será perdida. Mesmo se keepalives forem perdidos, o tronco permanecerá ativo até que cinco mensagens sequenciais sejam perdidas. Isso significa que uma conexão deve ficar inativa por 25 segundos antes que os troncos fiquem inativos.

Problemas comuns sobre truncamentos de conexão

Há vários bugs associados às conexões de tronco de voz. Verifique esses bugs se você vê algo incomum. Quando o Cisco IOS Software 12.2 foi lançado, a maioria desses problemas tinha sido abordada e integrada. Você pode examinar os bugs para se conscientizar de que essas são

causas de problemas com códigos mais antigos. Um dos problemas mais comuns é fazer com que os PBXs sinalizem corretamente sobre a conexão de tronco. Parece ser uma boa ideia desativar os troncos e configurar os roteadores para que funcionem em cada extremidade, mas a abordagem é realmente contraprodutiva, já que qualquer coisa que você mudou agora se torna raiz quando os troncos são estabelecidos. A melhor maneira de solucionar problemas é com os troncos ativos e funcionais.

Iniciar a solução de problemas

Énecessário examinar os fundamentos para estabelecer que essas funções funcionam corretamente:

- Os troncos estão estabelecidos? Emita o comando **show voice call summary** e verifique se os troncos estão no estado `S_CONNECTED`.
- Os DSPs estão processando pacotes? Emita o comando **show voice dsp** para verificar isso. Se você não vir pacotes processados pelos DSPs, é porque o VAD está ativado e suprime os pacotes. Desative o VAD, restabeleça os troncos e pesquise novamente. Além disso, verifique se os contadores de pacote aumentam quando o comando **show call ative voice brief** é emitido. Esse comando também mostra se o VAD está habilitado para o registro de chamada em questão.

Se os troncos se conectarem às portas analógicas em qualquer local, é melhor verificar a operação do PBX no modo não truncado. Para solucionar problemas de conectividade E&M analógica, consulte Entendendo e Troubleshooting de Tipos de Interface E&M Analógica e Disposições de Fiação . Depois que tudo for verificado e funcionar corretamente, ative os troncos e examine a sinalização transmitida entre os PBXs.

A maneira ideal de solucionar problemas de conexão de tronco de voz é examinar a sinalização transmitida entre os PBXs. É melhor ter uma sessão Telnet para cada roteador em questão para que a sinalização possa ser observada à medida que passa de uma extremidade à outra. Este documento usa sinalização de permissão E&M, pois é bastante popular e a temporização de permissão deve ser levada em consideração.

Esta é a saída do roteador conectado ao PBX que origina a chamada:

```
May 22 19:39:03.582: [3/0:0(1)] rcv from dsp sig DCBA state 0x0
!--- It is in idle state. May 22 19:39:07.774: [3/0:0(1)] send to dsp SIG DCBA state 0x0 !---
ABCD bits=0000. May 22 19:39:08.586: [3/0:0(1)] rcv from dsp SIG DCBA state 0x0 May 22
19:39:12.778: [3/0:0(1)] send to dsp SIG DCBA state 0x0 May 22 19:39:13.586: [3/0:0(1)] rcv from
dsp SIG DCBA state 0x0 May 22 19:39:17.777: [3/0:0(1)] send to dsp SIG DCBA state 0x0 May 22
19:39:18.593: [3/0:0(1)] rcv from dsp SIG DCBA state 0x0 May 22 19:39:22.781: [3/0:0(1)] send to
dsp SIG DCBA state 0x0 May 22 19:39:23.593: [3/0:0(1)] rcv from dsp SIG DCBA state 0x0 May 22
19:39:27.781: [3/0:0(1)] send to dsp SIG DCBA state 0x0 May 22 19:39:28.597: [3/0:0(1)] rcv from
dsp SIG DCBA state 0x0 May 22 19:39:32.785: [3/0:0(1)] send to dsp SIG DCBA state 0x0 May 22
19:39:33.597: [3/0:0(1)] rcv from dsp SIG DCBA state 0x0 May 22 19:39:37.789: [3/0:0(1)] send to
dsp SIG DCBA state 0x0 May 22 19:39:38.601: [3/0:0(1)] rcv from dsp SIG DCBA state 0x0 May 22
19:39:39.777: [3/0:0(1)] rcv from dsp SIG DCBA state 0x0 May 22 19:39:39.797: [3/0:0(1)] rcv
from dsp SIG DCBA state 0x0 May 22 19:39:39.817: [3/0:0(1)] rcv from dsp SIG DCBA state 0xF !---
Receives off-hook from PBX, and passes to remote end. May 22 19:39:39.837: [3/0:0(1)] rcv from
dsp SIG DCBA state 0xF May 22 19:39:39.857: [3/0:0(1)] rcv from dsp SIG DCBA state 0xF May 22
19:39:39.877: [3/0:0(1)] rcv from dsp SIG DCBA state 0xF May 22 19:39:39.897: [3/0:0(1)] rcv
from dsp SIG DCBA state 0xF May 22 19:39:39.917: [3/0:0(1)] rcv from dsp SIG DCBA state 0xF May
22 19:39:39.937: [3/0:0(1)] rcv from dsp SIG DCBA state 0xF May 22 19:39:39.957: [3/0:0(1)] rcv
from dsp SIG DCBA state 0xF May 22 19:39:39.977: [3/0:0(1)] rcv from dsp SIG DCBA state 0xF May
22 19:39:39.997: [3/0:0(1)] rcv from dsp SIG DCBA state 0xF May 22 19:39:40.017: [3/0:0(1)] rcv
```

from dsp SIG DCBA state 0xF May 22 19:39:40.037: [3/0:0(1)] rcv from dsp SIG DCBA state 0xF May 22 19:39:40.057: [3/0:0(1)] rcv from dsp SIG DCBA state 0xF May 22 19:39:40.077: [3/0:0(1)] rcv from dsp SIG DCBA state 0xF May 22 19:39:40.089: [3/0:0(1)] send to dsp SIG DCBA state 0x0 May 22 19:39:40.097: [3/0:0(1)] rcv from dsp SIG DCBA state 0xF May 22 19:39:40.109: [3/0:0(1)] send to dsp SIG DCBA state 0x0 May 22 19:39:40.117: [3/0:0(1)] rcv from dsp SIG DCBA state 0xF *!--- Receiving wink from remote side, and passes to PBX.* May 22 19:39:40.129: [3/0:0(1)] send to dsp SIG DCBA state 0xF May 22 19:39:40.137: [3/0:0(1)] rcv from dsp SIG DCBA state 0xF May 22 19:39:40.149: [3/0:0(1)] send to dsp SIG DCBA state 0xF May 22 19:39:40.157: [3/0:0(1)] rcv from dsp SIG DCBA state 0xF May 22 19:39:40.169: [3/0:0(1)] send to dsp SIG DCBA state 0xF May 22 19:39:40.177: [3/0:0(1)] rcv from dsp SIG DCBA state 0xF May 22 19:39:40.189: [3/0:0(1)] send to dsp SIG DCBA state 0xF May 22 19:39:40.197: [3/0:0(1)] rcv from dsp SIG DCBA state 0xF May 22 19:39:40.213: [3/0:0(1)] send to dsp SIG DCBA state 0xF May 22 19:39:40.217: [3/0:0(1)] rcv from dsp SIG DCBA state 0xF May 22 19:39:40.229: [3/0:0(1)] send to dsp SIG DCBA state 0xF May 22 19:39:40.237: [3/0:0(1)] rcv from dsp SIG DCBA state 0xF May 22 19:39:40.249: [3/0:0(1)] send to dsp SIG DCBA state 0xF May 22 19:39:40.257: [3/0:0(1)] rcv from dsp SIG DCBA state 0xF May 22 19:39:40.269: [3/0:0(1)] send to dsp SIG DCBA state 0xF May 22 19:39:40.289: [3/0:0(1)] send to dsp SIG DCBA state 0xF May 22 19:39:40.309: [3/0:0(1)] send to dsp SIG DCBA state 0xF May 22 19:39:40.329: [3/0:0(1)] send to dsp SIG DCBA state 0xF May 22 19:39:40.349: [3/0:0(1)] send to dsp SIG DCBA state 0x0 *!--- Wink ended from remote side, and passes to PBX.* May 22 19:39:40.369: [3/0:0(1)] send to dsp SIG DCBA state 0x0 May 22 19:39:40.389: [3/0:0(1)] send to dsp SIG DCBA state 0x0 May 22 19:39:40.409: [3/0:0(1)] send to dsp SIG DCBA state 0x0 May 22 19:39:40.429: [3/0:0(1)] send to dsp SIG DCBA state 0x0 May 22 19:39:40.449: [3/0:0(1)] send to dsp SIG DCBA state 0x0 May 22 19:39:40.469: [3/0:0(1)] send to dsp SIG DCBA state 0x0 May 22 19:39:40.493: [3/0:0(1)] send to dsp SIG DCBA state 0x0 May 22 19:39:40.509: [3/0:0(1)] send to dsp SIG DCBA state 0x0 May 22 19:39:40.529: [3/0:0(1)] send to dsp SIG DCBA state 0x0 May 22 19:39:40.549: [3/0:0(1)] send to dsp SIG DCBA state 0x0 May 22 19:39:40.569: [3/0:0(1)] send to dsp SIG DCBA state 0x0 May 22 19:39:40.589: [3/0:0(1)] send to dsp SIG DCBA state 0x0 May 22 19:39:40.613: [3/0:0(1)] send to dsp SIG DCBA state 0x0 May 22 19:39:40.629: [3/0:0(1)] send to dsp SIG DCBA state 0x0 May 22 19:39:40.649: [3/0:0(1)] send to dsp SIG DCBA state 0x0 May 22 19:39:40.669: [3/0:0(1)] send to dsp SIG DCBA state 0x0 May 22 19:39:40.689: [3/0:0(1)] send to dsp SIG DCBA state 0x0 May 22 19:39:40.709: [3/0:0(1)] send to dsp SIG DCBA state 0x0 May 22 19:39:40.729: [3/0:0(1)] send to dsp SIG DCBA state 0x0 May 22 19:39:40.749: [3/0:0(1)] send to dsp SIG DCBA state 0x0 May 22 19:39:40.769: [3/0:0(1)] send to dsp SIG DCBA state 0x0 May 22 19:39:45.773: [3/0:0(1)] send to dsp SIG DCBA state 0x0 May 22 19:39:50.081: [3/0:0(1)] send to dsp SIG DCBA state 0x0 May 22 19:39:50.101: [3/0:0(1)] send to dsp SIG DCBA state 0x0 May 22 19:39:50.121: [3/0:0(1)] send to dsp SIG DCBA state 0xF *!--- Wink ends, the remote end is now off-hook, the conversation happens.* May 22 19:39:50.141: [3/0:0(1)] send to dsp SIG DCBA state 0xF May 22 19:39:50.161: [3/0:0(1)] send to dsp SIG DCBA state 0xF May 22 19:39:50.181: [3/0:0(1)] send to dsp SIG DCBA state 0xF May 22 19:39:50.197: [3/0:0(1)] send to dsp SIG DCBA state 0xF May 22 19:39:50.221: [3/0:0(1)] send to dsp SIG DCBA state 0xF May 22 19:39:50.241: [3/0:0(1)] send to dsp SIG DCBA state 0xF May 22 19:39:50.261: [3/0:0(1)] send to dsp SIG DCBA state 0xF May 22 19:39:50.261: [3/0:0(1)] rcv from dsp SIG DCBA state 0xF May 22 19:39:50.281: [3/0:0(1)] send to dsp SIG DCBA state 0xF May 22 19:39:50.301: [3/0:0(1)] send to dsp SIG DCBA state 0xF May 22 19:39:50.321: [3/0:0(1)] send to dsp SIG DCBA state 0xF May 22 19:39:50.341: [3/0:0(1)] send to dsp SIG DCBA state 0xF May 22 19:39:50.361: [3/0:0(1)] send to dsp SIG DCBA state 0xF May 22 19:39:50.381: [3/0:0(1)] send to dsp SIG DCBA state 0xF May 22 19:39:50.401: [3/0:0(1)] send to dsp SIG DCBA state 0xF May 22 19:39:50.421: [3/0:0(1)] send to dsp SIG DCBA state 0xF May 22 19:39:50.441: [3/0:0(1)] send to dsp SIG DCBA state 0xF May 22 19:39:50.461: [3/0:0(1)] send to dsp SIG DCBA state 0xF May 22 19:39:50.481: [3/0:0(1)] send to dsp SIG DCBA state 0xF May 22 19:39:50.501: [3/0:0(1)] send to dsp SIG DCBA state 0xF May 22 19:39:50.521: [3/0:0(1)] send to dsp SIG DCBA state 0xF May 22 19:39:50.541: [3/0:0(1)] send to dsp SIG DCBA state 0xF May 22 19:39:50.561: [3/0:0(1)] send to dsp SIG DCBA state 0xF May 22 19:39:55.265: [3/0:0(1)] rcv from dsp SIG DCBA state 0xF May 22 19:39:55.561: [3/0:0(1)] send to dsp SIG DCBA state 0xF May 22 19:40:00.269: [3/0:0(1)] rcv from dsp SIG DCBA state 0xF May 22 19:40:00.565: [3/0:0(1)] send to dsp SIG DCBA state 0xF May 22 19:40:05.268: [3/0:0(1)] rcv from dsp SIG DCBA state 0xF May 22 19:40:05.564: [3/0:0(1)] send to dsp SIG DCBA state 0xF May 22 19:40:10.272: [3/0:0(1)] rcv from dsp SIG DCBA state 0xF May 22 19:40:10.568: [3/0:0(1)] send to dsp SIG DCBA state 0xF May 22 19:40:15.276: [3/0:0(1)] rcv from dsp SIG DCBA state 0xF May 22 19:40:15.572: [3/0:0(1)] send to dsp SIG DCBA state 0xF May 22 19:40:19.676: [3/0:0(1)] send to dsp SIG DCBA state 0xF May 22 19:40:19.696: [3/0:0(1)] send to dsp SIG DCBA state 0xF May 22 19:40:19.716: [3/0:0(1)] send to dsp SIG DCBA state 0x0 May 22 19:40:19.736: [3/0:0(1)] send to dsp SIG DCBA state 0x0 May 22 19:40:19.756: [3/0:0(1)] send to dsp SIG DCBA state 0x0 May 22 19:40:19.776: [3/0:0(1)] send to dsp SIG DCBA state 0x0 May 22 19:40:19.796: [3/0:0(1)] send to dsp SIG DCBA state 0x0 May 22 19:40:19.796: [3/0:0(1)] rcv from dsp SIG DCBA state 0xF May 22 19:40:19.816: [3/0:0(1)] send to

dsp SIG DCBA state 0x0 May 22 19:40:19.816: [3/0:0(1)] rcv from dsp SIG DCBA state 0xF May 22 19:40:19.836: [3/0:0(1)] send to dsp SIG DCBA state 0x0 May 22 19:40:19.836: [3/0:0(1)] rcv from dsp SIG DCBA state 0x0 *!--- Both side hung up, back to idle state.* May 22 19:40:19.856: [3/0:0(1)] send to dsp SIG DCBA state 0x0 May 22 19:40:19.856: [3/0:0(1)] rcv from dsp SIG DCBA state 0x0 May 22 19:40:19.876: [3/0:0(1)] send to dsp SIG DCBA state 0x0 May 22 19:40:19.876: [3/0:0(1)] rcv from dsp SIG DCBA state 0x0 May 22 19:40:19.896: [3/0:0(1)] send to dsp SIG DCBA state 0x0 May 22 19:40:19.896: [3/0:0(1)] rcv from dsp SIG DCBA state 0x0 May 22 19:40:19.916: [3/0:0(1)] send to dsp SIG DCBA state 0x0 May 22 19:40:19.916: [3/0:0(1)] rcv from dsp SIG DCBA state 0x0 May 22 19:40:19.936: [3/0:0(1)] send to dsp SIG DCBA state 0x0

Esta saída mostra que o roteador encerra a chamada. O NTP (Protocolo de tempo de rede) foi sincronizado.

May 22 19:39:03.582: [3/0:0(1)] send to dsp SIG DCBA state 0x0
May 22 19:39:07.774: [3/0:0(1)] rcv from dsp SIG DCBA state 0x0
!--- Idle state, both side on-hook. May 22 19:39:08.586: [3/0:0(1)] send to dsp SIG DCBA state 0x0 May 22 19:39:12.774: [3/0:0(1)] rcv from dsp SIG DCBA state 0x0 May 22 19:39:13.586: [3/0:0(1)] send to dsp SIG DCBA state 0x0 May 22 19:39:15.383: [1/0:0(1)] Signaling RTP packet has no particle *!--- You will see this message if you are running Cisco IOS !--- Software Release 12.2(1a) or later. It is not an error !--- message, it is a normal functioning state.* May 22 19:39:17.774: [3/0:0(1)] rcv from dsp SIG DCBA state 0x0 May 22 19:39:18.590: [3/0:0(1)] send to dsp SIG DCBA state 0x0 May 22 19:39:22.778: [3/0:0(1)] rcv from dsp SIG DCBA state 0x0 May 22 19:39:23.594: [3/0:0(1)] send to dsp SIG DCBA state 0x0 May 22 19:39:27.782: [3/0:0(1)] rcv from dsp SIG DCBA state 0x0 May 22 19:39:28.598: [3/0:0(1)] send to dsp SIG DCBA state 0x0 May 22 19:39:32.782: [3/0:0(1)] rcv from dsp SIG DCBA state 0x0 May 22 19:39:33.598: [3/0:0(1)] send to dsp SIG DCBA state 0x0 May 22 19:39:37.786: [3/0:0(1)] rcv from dsp SIG DCBA state 0x0 May 22 19:39:38.602: [3/0:0(1)] send to dsp SIG DCBA state 0x0 May 22 19:39:39.778: [3/0:0(1)] send to dsp SIG DCBA state 0x0 May 22 19:39:39.798: [3/0:0(1)] send to dsp SIG DCBA state 0x0 May 22 19:39:39.818: [3/0:0(1)] send to dsp SIG DCBA state 0xF *!--- Remote side off-hook, this is conveyed to the PBX.* May 22 19:39:39.838: [3/0:0(1)] send to dsp SIG DCBA state 0xF May 22 19:39:39.858: [3/0:0(1)] send to dsp SIG DCBA state 0xF May 22 19:39:39.878: [3/0:0(1)] send to dsp SIG DCBA state 0xF May 22 19:39:39.898: [3/0:0(1)] send to dsp SIG DCBA state 0xF May 22 19:39:39.918: [3/0:0(1)] send to dsp SIG DCBA state 0xF May 22 19:39:39.938: [3/0:0(1)] send to dsp SIG DCBA state 0xF May 22 19:39:39.958: [3/0:0(1)] send to dsp SIG DCBA state 0xF May 22 19:39:39.978: [3/0:0(1)] send to dsp SIG DCBA state 0xF May 22 19:39:39.998: [3/0:0(1)] send to dsp SIG DCBA state 0xF May 22 19:39:40.018: [3/0:0(1)] send to dsp SIG DCBA state 0xF May 22 19:39:40.038: [3/0:0(1)] send to dsp SIG DCBA state 0xF May 22 19:39:40.058: [3/0:0(1)] send to dsp SIG DCBA state 0xF May 22 19:39:40.078: [3/0:0(1)] send to dsp SIG DCBA state 0xF May 22 19:39:40.090: [3/0:0(1)] rcv from dsp SIG DCBA state 0x0 May 22 19:39:40.098: [3/0:0(1)] send to dsp SIG DCBA state 0xF May 22 19:39:40.110: [3/0:0(1)] rcv from dsp SIG DCBA state 0x0 May 22 19:39:40.118: [3/0:0(1)] send to dsp SIG DCBA state 0xF May 22 19:39:40.130: [3/0:0(1)] rcv from dsp SIG DCBA state 0xF *!--- Receive wink from PBX.* May 22 19:39:40.138: [3/0:0(1)] send to dsp SIG DCBA state 0xF May 22 19:39:40.150: [3/0:0(1)] rcv from dsp SIG DCBA state 0xF May 22 19:39:40.158: [3/0:0(1)] send to dsp SIG DCBA state 0xF May 22 19:39:40.170: [3/0:0(1)] rcv from dsp SIG DCBA state 0xF May 22 19:39:40.178: [3/0:0(1)] send to dsp SIG DCBA state 0xF May 22 19:39:40.190: [3/0:0(1)] rcv from dsp SIG DCBA state 0xF May 22 19:39:40.198: [3/0:0(1)] send to dsp SIG DCBA state 0xF May 22 19:39:40.210: [3/0:0(1)] rcv from dsp SIG DCBA state 0xF May 22 19:39:40.218: [3/0:0(1)] send to dsp SIG DCBA state 0xF May 22 19:39:40.230: [3/0:0(1)] rcv from dsp SIG DCBA state 0xF May 22 19:39:40.238: [3/0:0(1)] send to dsp SIG DCBA state 0xF May 22 19:39:40.250: [3/0:0(1)] rcv from dsp SIG DCBA state 0xF May 22 19:39:40.258: [3/0:0(1)] send to dsp SIG DCBA state 0xF May 22 19:39:40.270: [3/0:0(1)] rcv from dsp SIG DCBA state 0xF May 22 19:39:40.290: [3/0:0(1)] rcv from dsp SIG DCBA state 0xF May 22 19:39:40.310: [3/0:0(1)] rcv from dsp SIG DCBA state 0xF May 22 19:39:40.330: [3/0:0(1)] rcv from dsp SIG DCBA state 0xF May 22 19:39:40.350: [3/0:0(1)] rcv from dsp SIG DCBA state 0x0 *!--- Wink ended, waiting for an answer.* May 22 19:39:40.370: [3/0:0(1)] rcv from dsp SIG DCBA state 0x0 May 22 19:39:40.390: [3/0:0(1)] rcv from dsp SIG DCBA state 0x0 May 22 19:39:40.410: [3/0:0(1)] rcv from dsp SIG DCBA state 0x0 May 22 19:39:40.430: [3/0:0(1)] rcv from dsp SIG DCBA state 0x0 May 22 19:39:40.450: [3/0:0(1)] rcv from dsp SIG DCBA state 0x0 May 22 19:39:40.470: [3/0:0(1)] rcv from dsp SIG DCBA state 0x0 May 22 19:39:40.490: [3/0:0(1)] rcv from dsp SIG DCBA state 0x0 May 22 19:39:40.510: [3/0:0(1)] rcv from dsp SIG DCBA state 0x0 May 22 19:39:40.530: [3/0:0(1)] rcv from dsp SIG DCBA state 0x0 May 22 19:39:40.550: [3/0:0(1)] rcv from dsp SIG DCBA state 0x0 May 22 19:39:40.570: [3/0:0(1)] rcv from dsp SIG DCBA state 0x0 May 22 19:39:40.590: [3/0:0(1)] rcv from dsp SIG DCBA state 0x0 May 22 19:39:40.610: [3/0:0(1)] rcv from dsp SIG DCBA state 0x0 May 22 19:39:40.630:

```
[3/0:0(1)] rcv from dsp SIG DCBA state 0x0 May 22 19:39:40.650: [3/0:0(1)] rcv from dsp SIG DCBA
state 0x0 May 22 19:39:40.670: [3/0:0(1)] rcv from dsp SIG DCBA state 0x0 May 22 19:39:40.690:
[3/0:0(1)] rcv from dsp SIG DCBA state 0x0 May 22 19:39:40.710: [3/0:0(1)] rcv from dsp SIG DCBA
state 0x0 May 22 19:39:40.730: [3/0:0(1)] rcv from dsp SIG DCBA state 0x0 May 22 19:39:40.750:
[3/0:0(1)] rcv from dsp SIG DCBA state 0x0 May 22 19:39:40.770: [3/0:0(1)] rcv from dsp SIG DCBA
state 0x0 May 22 19:39:45.262: [3/0:0(1)] send to dsp SIG DCBA state 0xF May 22 19:39:45.770:
[3/0:0(1)] rcv from dsp SIG DCBA state 0x0 May 22 19:39:50.077: [3/0:0(1)] rcv from dsp SIG DCBA
state 0x0 May 22 19:39:50.097: [3/0:0(1)] rcv from dsp SIG DCBA state 0x0 May 22 19:39:50.117:
[3/0:0(1)] rcv from dsp SIG DCBA state 0xF !--- Receive off-hook from PBX. May 22 19:39:50.137:
[3/0:0(1)] rcv from dsp SIG DCBA state 0xF May 22 19:39:50.157: [3/0:0(1)] rcv from dsp SIG DCBA
state 0xF May 22 19:39:50.177: [3/0:0(1)] rcv from dsp SIG DCBA state 0xF May 22 19:39:50.197:
[3/0:0(1)] rcv from dsp SIG DCBA state 0xF May 22 19:39:50.217: [3/0:0(1)] rcv from dsp SIG DCBA
state 0xF May 22 19:39:50.237: [3/0:0(1)] rcv from dsp SIG DCBA state 0xF May 22 19:39:50.257:
[3/0:0(1)] rcv from dsp SIG DCBA state 0xF May 22 19:39:50.261: [3/0:0(1)] send to dsp SIG DCBA
state 0xF May 22 19:39:50.277: [3/0:0(1)] rcv from dsp SIG DCBA state 0xF May 22 19:39:50.297:
[3/0:0(1)] rcv from dsp SIG DCBA state 0xF May 22 19:39:50.317: [3/0:0(1)] rcv from dsp SIG DCBA
state 0xF May 22 19:39:50.337: [3/0:0(1)] rcv from dsp SIG DCBA state 0xF May 22 19:39:50.357:
[3/0:0(1)] rcv from dsp SIG DCBA state 0xF May 22 19:39:50.377: [3/0:0(1)] rcv from dsp SIG DCBA
state 0xF May 22 19:39:50.397: [3/0:0(1)] rcv from dsp SIG DCBA state 0xF May 22 19:39:50.417:
[3/0:0(1)] rcv from dsp SIG DCBA state 0xF May 22 19:39:50.437: [3/0:0(1)] rcv from dsp SIG DCBA
state 0xF May 22 19:39:50.457: [3/0:0(1)] rcv from dsp SIG DCBA state 0xF May 22 19:39:50.477:
[3/0:0(1)] rcv from dsp SIG DCBA state 0xF May 22 19:39:50.497: [3/0:0(1)] rcv from dsp SIG DCBA
state 0xF May 22 19:39:50.517: [3/0:0(1)] rcv from dsp SIG DCBA state 0xF May 22 19:39:50.537:
[3/0:0(1)] rcv from dsp SIG DCBA state 0xF May 22 19:39:50.557: [3/0:0(1)] rcv from dsp SIG DCBA
state 0xF !--- Both sides off-hook, the conversation happens. May 22 19:39:55.265: [3/0:0(1)]
send to dsp SIG DCBA state 0xF May 22 19:39:55.557: [3/0:0(1)] rcv from dsp SIG DCBA state 0xF
May 22 19:40:00.269: [3/0:0(1)] send to dsp SIG DCBA state 0xF May 22 19:40:00.561: [3/0:0(1)]
rcv from dsp SIG DCBA state 0xF May 22 19:40:05.269: [3/0:0(1)] send to dsp SIG DCBA state 0xF
May 22 19:40:05.561: [3/0:0(1)] rcv from dsp SIG DCBA state 0xF May 22 19:40:10.273: [3/0:0(1)]
send to dsp SIG DCBA state 0xF May 22 19:40:10.565: [3/0:0(1)] rcv from dsp SIG DCBA state 0xF
May 22 19:40:15.273: [3/0:0(1)] send to dsp SIG DCBA state 0xF May 22 19:40:15.569: [3/0:0(1)]
rcv from dsp SIG DCBA state 0xF May 22 19:40:19.673: [3/0:0(1)] rcv from dsp SIG DCBA state 0xF
May 22 19:40:19.693: [3/0:0(1)] rcv from dsp SIG DCBA state 0xF May 22 19:40:19.713: [3/0:0(1)]
rcv from dsp SIG DCBA state 0x0 May 22 19:40:19.733: [3/0:0(1)] rcv from dsp SIG DCBA state 0x0
May 22 19:40:19.753: [3/0:0(1)] rcv from dsp SIG DCBA state 0x0 May 22 19:40:19.773: [3/0:0(1)]
rcv from dsp SIG DCBA state 0x0 May 22 19:40:19.793: [3/0:0(1)] rcv from dsp SIG DCBA state 0x0
May 22 19:40:19.797: [3/0:0(1)] send to dsp SIG DCBA state 0xF May 22 19:40:19.813: [3/0:0(1)]
rcv from dsp SIG DCBA state 0x0 May 22 19:40:19.817: [3/0:0(1)] send to dsp SIG DCBA state 0xF
May 22 19:40:19.833: [3/0:0(1)] rcv from dsp SIG DCBA state 0x0 May 22 19:40:19.837: [3/0:0(1)]
send to dsp SIG DCBA state 0x0 !--- Both sides are back on-hook, back to idle. May 22
19:40:19.853: [3/0:0(1)] rcv from dsp SIG DCBA state 0x0 May 22 19:40:19.857: [3/0:0(1)] send to
dsp SIG DCBA state 0x0 May 22 19:40:19.873: [3/0:0(1)] rcv from dsp SIG DCBA state 0x0 May 22
19:40:19.877: [3/0:0(1)] send to dsp SIG DCBA state 0x0 May 22 19:40:19.893: [3/0:0(1)] rcv from
dsp SIG DCBA state 0x0 May 22 19:40:19.897: [3/0:0(1)] send to dsp SIG DCBA state 0x0
```

Observação: essa saída mostra a sinalização que ocorre em ambos os lados de um tronco de voz que usa sinalização de trunk E&M. Outros tipos de sinalização podem ser vistos que usam essas mesmas depurações. Se as chamadas forem estabelecidas corretamente (como mostrado aqui), deverá haver um áudio bidirecional. Isso pode ser verificado se você observar a saída do comando **show voice dsp** ou **show call ative voice brief**. Se tudo parecer estar bem lá e você estiver tendo problemas de áudio (sem áudio ou unidirecional) com conexões analógicas, verifique essas conexões novamente.

Determine quais chamadas estão ativas

Como não serve para ver a saída do comando **show call ative voice** ou **show voice call summary** para chamadas em tronco, você precisa de um método simples para determinar quais troncos de voz suportam chamadas ativas. Uma das maneiras mais fáceis de fazer isso é emitir o comando **show voice trunk-Condialing** em conjunto com o parâmetro *include* e usar *ABCD* como a string incluída, como mostrado aqui:

```
Phoenix#show voice trunk-conditioning signaling | include ABCD
last-TX-ABCD=0000, last-RX-ABCD=0000
last-TX-ABCD=0000, last-RX-ABCD=0000
last-TX-ABCD=0000, last-RX-ABCD=0000
last-TX-ABCD=0000, last-RX-ABCD=0000
last-TX-ABCD=0000, last-RX-ABCD=0000
last-TX-ABCD=0000, last-RX-ABCD=0000
last-TX-ABCD=0000, last-RX-ABCD=0000
last-TX-ABCD=1111, last-RX-ABCD=0000
!--- Timeslot 8. last-TX-ABCD=0000, last-RX-ABCD=0000 last-TX-ABCD=1111, last-RX-ABCD=1111 !---
Timeslot 10. last-TX-ABCD=0000, last-RX-ABCD=0000 last-TX-ABCD=0000, last-RX-ABCD=0000 last-TX-
ABCD=0000, last-RX-ABCD=0000 last-TX-ABCD=0000, last-RX-ABCD=0000 last-TX-ABCD=0000, last-RX-
ABCD=0000 last-TX-ABCD=0000, last-RX-ABCD=0000
```

Observação: esta saída mostra uma chamada ativa no timeslot 10 e outra chamada sendo iniciada no timeslot oito. Você deseja criar um alias para esse comando bastante longo se usá-lo muito.

Solução de problemas de DTMF

Além da sinalização fora do gancho e no gancho, a única outra coisa que os roteadores passam entre os PBXs (além da voz) são tons DTMF. Também há um caminho de áudio, portanto, isso geralmente não é um problema, mas há um problema. O problema surge com a forma como você faz áudio por esse caminho. Às vezes, é preferível usar os codecs de taxa de bits baixa para economizar largura de banda. A questão é que esses codecs de baixa taxa de bits são projetados por meio de algoritmos que foram escritos para a fala humana. Os tons de DTMF não estão em conformidade com esses algoritmos muito bem e precisam de algum outro método para transmitir, a menos que o cliente use o codec g711. A resposta está no comando **dtmf-relay**. Esse recurso permite que os DSPs no final, inicie o tom, reconheça o tom DTMF e o separe do fluxo de áudio regular. Com base em como ele é configurado, o DSP codifica esse tom como um tipo diferente de pacote do protocolo de tempo real (RTP) ou como uma mensagem h245 a ser enviada pelo link separadamente do fluxo de áudio. Esse é o mesmo processo por trás dos comandos fax-relay e modem-relay.

Esse recurso apresenta outro problema de depuração para a solução de problemas de tronco. Como você verifica quais dígitos são passados se não há configuração de chamada e precisa extrair essas informações do fluxo de pacotes entre os roteadores? Como fazer isso depende do tipo de comando **dtmf-relay** usado.

Como mostrado neste exemplo, o comando **dtmf-relay cisco-rtp** usa um tipo de payload proprietário da Cisco, então você deve olhar para os DSPs para ver isso. Você pode executar o comando **debug vpm signal** em conjunto com o comando **debug vpm port x/x:y.z** (para limitar a saída à porta em questão) para ver os dígitos passados aos DSPs no lado de origem. Essa saída é exibida no lado de origem, não no lado de terminação.

```
*Mar  1 00:22:39.592: htsp_digit_ready: digit = 31
*Mar  1 00:22:39.592: [1/0:1(2), S_TRUNKED, E_VTSP_DIGIT]
*Mar  1 00:22:40.021: htsp_digit_ready: digit = 32
*Mar  1 00:22:40.021: [1/0:1(2), S_TRUNKED, E_VTSP_DIGIT]
*Mar  1 00:22:40.562: htsp_digit_ready: digit = 33
*Mar  1 00:22:40.562: [1/0:1(2), S_TRUNKED, E_VTSP_DIGIT]
*Mar  1 00:22:40.810: [1/0:1(2)] rcv from dsp SIG DCBA state 0xF
*Mar  1 00:22:41.131: htsp_digit_ready: digit = 34
*Mar  1 00:22:41.131: [1/0:1(2), S_TRUNKED, E_VTSP_DIGIT]
*Mar  1 00:22:41.499:  [1/0:1(2)] Signaling RTP packet has no partical
```

```
*Mar  1 00:22:41.499: [1/0:1(2)] send to dsp SIG DCBA state 0xF
*Mar  1 00:22:41.672: htsp_digit_ready: digit = 35
*Mar  1 00:22:41.672: [1/0:1(2), S_TRUNKED, E_VTSP_DIGIT]
*Mar  1 00:22:42.192: htsp_digit_ready: digit = 36
*Mar  1 00:22:42.192: [1/0:1(2), S_TRUNKED, E_VTSP_DIGIT]
*Mar  1 00:22:42.789: htsp_digit_ready: digit = 37
*Mar  1 00:22:42.789: [1/0:1(2), S_TRUNKED, E_VTSP_DIGIT]
*Mar  1 00:22:43.350: htsp_digit_ready: digit = 38
*Mar  1 00:22:43.350: [1/0:1(2), S_TRUNKED, E_VTSP_DIGIT]
*Mar  1 00:22:44.079: htsp_digit_ready: digit = 39
*Mar  1 00:22:44.079: [1/0:1(2), S_TRUNKED, E_VTSP_DIGIT]
*Mar  1 00:22:45.249: htsp_digit_ready: digit = 30
*Mar  1 00:22:45.249: [1/0:1(2), S_TRUNKED, E_VTSP_DIGIT]
*Mar  1 00:22:45.810: [1/0:1(2)] rcv from dsp SIG DCBA state 0xF
*Mar  1 00:22:46.007: htsp_digit_ready: digit = 2A
*Mar  1 00:22:46.011: [1/0:1(2), S_TRUNKED, E_VTSP_DIGIT]
*Mar  1 00:22:46.572: [1/0:1(2)] Signaling RTP packet has no partical
*Mar  1 00:22:46.572: [1/0:1(2)] send to dsp SIG DCBA state 0xF
*Mar  1 00:22:46.628: htsp_digit_ready: digit = 23
*Mar  1 00:22:46.628: [1/0:1(2), S_TRUNKED, E_VTSP_DIGIT]
*Mar  1 00:22:50.815: [1/0:1(2)] rcv from dsp SIG DCBA state 0xF
all digits 0-9 are represented by 30-39, * = 2A and # = 23.
```

Você pode verificar quais dígitos são enviados do lado de origem com o comando dtmf-relay h245-alfanumérico. O **comando dtmf-relay h245-alfanumérico** usa a parte alfanumérica do h.245 para transmitir os tons. Como mostrado neste exemplo, os dígitos podem ser facilmente vistos nos lados de origem e de terminação do tronco quando o comando **debug h245 asn1** está ativado:

Lado de origem:

```
*Mar  1 00:34:17.749: H245 MSC OUTGOING PDU ::=
value MultimediaSystemControlMessage ::= indication : userInput : alphanumeric : "1"

*Mar  1 00:34:17.749: H245 MSC OUTGOING ENCODE BUFFER::= 6D 400131
*Mar  1 00:34:17.753:
*Mar  1 00:34:18.350: H245 MSC OUTGOING PDU ::=
value MultimediaSystemControlMessage ::= indication : userInput : alphanumeric : "2"

*Mar  1 00:34:18.350: H245 MSC OUTGOING ENCODE BUFFER::= 6D 400132
*Mar  1 00:34:18.350:
*Mar  1 00:34:18.838: H245 MSC OUTGOING PDU ::=
value MultimediaSystemControlMessage ::= indication : userInput : alphanumeric : "3"

*Mar  1 00:34:18.838: H245 MSC OUTGOING ENCODE BUFFER::= 6D 400133
```

Lado de terminação:

```
*Mar  1 17:45:16.424: H245 MSC INCOMING ENCODE BUFFER::= 6D 400131
*Mar  1 17:45:16.424:
*Mar  1 17:45:16.424: H245 MSC INCOMING PDU ::=
value MultimediaSystemControlMessage ::= indication : userInput : alphanumeric : "1"

*Mar  1 17:45:17.025: H245 MSC INCOMING ENCODE BUFFER::= 6D 400132
*Mar  1 17:45:17.025:
*Mar  1 17:45:17.025: H245 MSC INCOMING PDU ::=
value MultimediaSystemControlMessage ::= indication : userInput : alphanumeric : "2"

*Mar  1 17:45:17.514: H245 MSC INCOMING ENCODE BUFFER::= 6D 400133
*Mar  1 17:45:17.514:
*Mar  1 17:45:17.514: H245 MSC INCOMING PDU ::=
```

```
value MultimediaSystemControlMessage ::= indication : userInput : alphanumeric : "3"
```

O comando dtmf-relay h245-signal é muito semelhante e pode ser visto quando são usadas as mesmas depurações que o comando **dtmf-relay h245-alfanumérico**. Em geral, solucionar problemas dos troncos de conexão com o comando **dtmf-relay** é bastante difícil sem as depurações mencionadas.

Informações Relacionadas

- Configurando e Troubleshooting de Transparent CCS
- Suporte à Tecnologia de Voz
- Suporte aos produtos de Voz e Comunicação por IP
- Troubleshooting da Telefonia IP Cisco
- Suporte Técnico - Cisco Systems