Avaya Aura® Call Center Elite Call Vectoring Feature Description

Release 10.2.x
Issue 1
December 2023

Notice

While reasonable efforts have been made to ensure that the information in this document is complete and accurate at the time of printing, Avaya assumes no liability for any errors. Avaya reserves the right to make changes and corrections to the information in this document without the obligation to notify any person or organization of such changes.

Documentation disclaimer

"Documentation" means information published in varying media which may include product information, subscription or service descriptions, operating instructions and performance specifications that are generally made available to users of products. Documentation does not include marketing materials. Avaya shall not be responsible for any modifications, additions, or deletions to the original published version of Documentation unless such modifications, additions, or deletions were performed by or on the express behalf of Avaya. End user agrees to indemnify and hold harmless Avaya, Avaya's agents, servants and employees against all claims, lawsuits, demands and judgments arising out of, or in connection with, subsequent modifications, additions or deletions to this documentation, to the extent made by End user.

Link disclaimer

Avaya is not responsible for the contents or reliability of any linked websites referenced within this site or Documentation provided by Avaya. Avaya is not responsible for the accuracy of any information, statement or content provided on these sites and does not necessarily endorse the products, services, or information described or offered within them. Avaya does not guarantee that these links will work all the time and has no control over the availability of the linked pages.

Warranty

Avaya provides a limited warranty on Avaya hardware and software. Please refer to your agreement with Avaya to establish the terms of the limited warranty. In addition, Avaya's standard warranty language as well as information regarding support for this product while under warranty is available to Avaya customers and other parties through the Avaya Support website: https://support.avaya.com/helpcenter/getGenericDetails?detailId=C20091120112456651010 under the link "Warranty & Product Lifecycle" or such successor site as designated by Avaya. Please note that if the product(s) was purchased from an authorized Avaya channel partner outside of the United States and Canada, the warranty is provided by said Avaya Channel Partner and not by Avaya.

"**Hosted Service**" means an Avaya hosted service subscription that You acquire from either Avaya or an authorized Avaya Channel Partner (as applicable) and which is described further in Hosted SAS or other service description documentation regarding the applicable hosted service. If You purchase a Hosted Service subscription, the foregoing limited warranty may not apply but You may be entitled to support services in connection with the Hosted Service as described further in your service description documents for the applicable Hosted Service. Contact Avaya or Avaya Channel Partner (as applicable) for more information.

Hosted Service

THE FOLLOWING APPLIES ONLY IF YOU PURCHASE AN AVAYA HOSTED SERVICE SUBSCRIPTION FROM AVAYA OR AN AVAYA CHANNEL PARTNER (AS APPLICABLE), THE TERMS OF USE FOR HOSTED SERVICES ARE AVAILABLE ON THE AVAYA WEBSITE, HTTPS://SUPPORT.AVAYA.COM/LICENSEINFO UNDER THE LINK "Avaya Terms of Use for Hosted Services" OR SUCH SUCCESSOR SITE AS DESIGNATED BY AVAYA, AND ARE APPLICABLE TO ANYONE WHO ACCESSES OR USES THE HOSTED SERVICE. BY ACCESSING OR USING THE HOSTED SERVICE, OR AUTHORIZING OTHERS TO DO SO, YOU, ON BEHALF OF YOURSELF AND THE ENTITY FOR WHOM YOU ARE DOING SO (HEREINAFTER REFERRED TO INTERCHANGEABLY AS "YOU" AND "END USER"), AGREE TO THE TERMS OF USE. IF YOU ARE ACCEPTING THE TERMS OF USE ON BEHALF A COMPANY OR OTHER LEGAL ENTITY, YOU REPRESENT THAT YOU HAVE THE AUTHORITY TO BIND SUCH ENTITY TO THESE TERMS OF USE. IF YOU DO NOT HAVE SUCH AUTHORITY, OR IF YOU DO NOT WISH TO ACCEPT THESE TERMS OF USE, YOU MUST NOT ACCESS OR USE THE HOSTED SERVICE OR AUTHORIZE ANYONE TO ACCESS OR USE THE HOSTED SERVICE.

Licenses

The Global Software License Terms ("Software License Terms") are available on the following website https://www.avaya.com/en/legal-license-terms/ or any successor site as designated by Avaya. These Software License Terms are applicable to anyone who installs, downloads, and/or uses Software and/or Documentation. By installing, downloading or using the Software, or authorizing others to do so, the end user agrees that the Software License Terms create a binding contract between them and Avaya. In case the end user is accepting these Software License Terms on behalf of a company or other legal entity, the end user represents that it has the authority to bind such entity to these Software License Terms.

Copyright

Except where expressly stated otherwise, no use should be made of materials on this site, the Documentation, Software, Hosted Service, or hardware provided by Avaya. All content on this site, the documentation, Hosted Service, and the product provided by Avaya including the selection, arrangement and design of the content is owned either by Avaya or its licensors and is protected by copyright and other intellectual property laws including the sui generis rights relating to the protection of databases. You may not modify, copy, reproduce, republish, upload, post, transmit or distribute in any way any content, in whole or in part, including any code and software unless expressly authorized by Avaya. Unauthorized reproduction, transmission, dissemination, storage, or use without the express written consent of Avaya can be a criminal, as well as a civil offense under the applicable law.

Virtualization

The following applies if the product is deployed on a virtual machine. Each product has its own ordering code and license types. Unless otherwise stated, each Instance of a product must be separately licensed and ordered. For example, if the end user customer or Avaya Channel Partner would like to install two Instances of the same type of products, then two products of that type must be ordered.

Third Party Components

The following applies only if the H.264 (AVC) codec is distributed with the product. THIS PRODUCT IS LICENSED UNDER THE AVC PATENT PORTFOLIO LICENSE FOR THE PERSONAL USE OF A CONSUMER OR OTHER USES IN WHICH IT DOES NOT RECEIVE REMUNERATION TO (i) ENCODE VIDEO IN COMPLIANCE WITH THE AVC STANDARD ("AVC VIDEO") AND/OR (ii) DECODE AVC VIDEO THAT WAS ENCODED BY A CONSUMER ENGAGED IN A PERSONAL ACTIVITY AND/OR WAS OBTAINED FROM A VIDEO PROVIDER LICENSED TO PROVIDE AVC VIDEO. NO LICENSE IS GRANTED OR SHALL BE IMPLIED FOR ANY OTHER USE. ADDITIONAL INFORMATION MAY BE OBTAINED FROM MPEG LA, L.L.C. SEE HTTP://WWW.MPEGLA.COM.

Service Provider

WITH RESPECT TO CODECS, IF THE AVAYA CHANNEL PARTNER IS HOSTING ANY PRODUCTS THAT USE OR EMBED THE H.264 CODEC OR H.265 CODEC, THE AVAYA CHANNEL PARTNER ACKNOWLEDGES AND AGREES THE AVAYA CHANNEL PARTNER IS RESPONSIBLE FOR ANY AND ALL RELATED FEES AND/OR ROYALTIES. THE H.264 (AVC) CODEC IS LICENSED UNDER THE AVC PATENT PORTFOLIO LICENSE FOR THE PERSONAL USE OF A CONSUMER OR OTHER USES IN WHICH IT DOES NOT RECEIVE REMUNERATION TO: (i) ENCODE VIDEO IN COMPLIANCE WITH THE AVC STANDARD ("AVC VIDEO") AND/OR (ii) DECODE AVC VIDEO THAT WAS ENCODED BY A CONSUMER ENGAGED IN A PERSONAL ACTIVITY AND/OR WAS OBTAINED FROM A VIDEO PROVIDER LICENSED TO PROVIDE AVC VIDEO. NO LICENSE IS GRANTED OR SHALL BE IMPLIED FOR ANY OTHER USE. ADDITIONAL INFORMATION FOR H.264 (AVC) AND H.265 (HEVC) CODECS MAY BE OBTAINED FROM MPEG LA, L.L.C. SEE HTTP://WWW.MPEGLA.COM.

Compliance with Laws

You acknowledge and agree that it is Your responsibility to comply with any applicable laws and regulations, including, but not limited to laws and regulations related to call recording, data privacy, intellectual property, trade secret, fraud, and music performance rights, in the country or territory where the Avaya product is used.

Preventing Toll Fraud

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

Avaya Toll Fraud intervention

If You suspect that You are being victimized by Toll Fraud and You need technical assistance or support, please contact your Avaya Sales Representative.

Security Vulnerabilities

Information about Avaya's security support policies can be found in the Security Policies and Support section of https://support.avaya.com/security.

Suspected Avaya product security vulnerabilities are handled per the Avaya Product Security Support Flow (https://support.avaya.com/css/P8/documents/100161515).

Downloading Documentation

For the most current versions of Documentation, see the Avaya Support website: https://support.avaya.com, or such successor site as designated by Avaya.

Contact Avaya Support

See the Avaya Support website: https://support.avaya.com for Product or Cloud Service notices and articles, or to report a problem with your Avaya Product or Cloud Service. For a list of support telephone numbers and contact addresses, go to the Avaya Support website: https://support.avaya.com (or such successor site as designated by Avaya), scroll to the bottom of the page, and select Contact Avaya Support.

Trademarks

All non-Avaya trademarks are the property of their respective owners.

Contents

Chapter 1: Introduction

Purpose

This document contains Avaya Aura® Call Center Elite high-level feature descriptions for Call Vectoring and provides details about feature characteristics, capabilities, capacities and interactions.

This document is intended for people who want a high-level understanding of Avaya Aura® Call Center Elite Call Vectoring features and capabilities.

Chapter 2: Call Vectoring features

Adjunct Routing

Adjunct Routing provides a means for an Adjunct Switch Application Interface (ASAI) processor to specify the destination of an arriving call when the processor encounters an **`adjunct routing link`** vector command during vector processing. An adjunct is a processor connected to Communication Manager that uses the ASAI protocol. The adjunct makes routing decisions based on caller information or agent availability, and returns a call route response to Communication Manager.

Communication Manager provides information in an ASAI route request message. Adjunct applications use the message to access a database and determine a route for the call. In a typical application, an ASAI adjunct uses the dialed number, Calling Party Number/Billing Number (CPN/BN), or the digits collected by Call Prompting to access caller information and to determine an appropriate call route.

Adjunct routing can be used in conjunction with the Call Prompting and Look-Ahead Interflow (LAI). When combined with one of those features, the following rules apply:

- When combined with Call Prompting, adjunct routing can pass up to 16 digits that are collected from the last relevant **`collect digits`** vector command.
- When combined with LAI, adjunct routing can pass the LAI information element or other call center-related data (with enhanced Information Forwarding) that was passed from the originating Communication Manager in the message or associated with the call from the local Communication Manager.

Adjunct Routing considerations

Consider the following points when working with Adjunct Routing:

- Before vector steps, try to insert a brief wait step for vector processing to handle any possible delays that cause the vector to fail and the call not to be routed. Follow this best practice when programming a vector:
    - Start all vectors with a “wait-time 0 seconds” step. Adding this step helps ensure you will have a touch-tone receiver (TTR) for vector processing.
    - When using an “adjunct routing” step, try to put a “wait-time 0 seconds” step before the “adjunct routing” step to provide a pause before routing the call to the adjunct system.

  - Any time digits are collected from a caller using a “collect digits” step, the step should be preceded by a “wait-time 0 seconds” step, hearing either ringback or music. This ensures you have a TTR to collect those digits.
- An adjunct specified in an **`adjunct routing link`** command can route a call to an internal number, an external number, a split, a VDN, an announcement extension, or to a particular agent. An adjunct can also provide priority ringing, priority queuing, and specify that a route-to an agent be done as a direct agent call.
- You can also include more than two consecutive adjunct routing link steps in a vector. This approach offers the following advantages:
  - Redundancy for ASAI link or application failure.
  - Simultaneous processing of multiple route requests, which distributes incoming call load more efficiently and results in faster call processing times.
- Vector processing continues to occur while an ASAI route request is being processed. Consequently, the first step to follow more than one adjunct routing link steps must be an announcement or a wait time step that adheres to the following rules:
  - If an **`announcement`** step follows immediately after an **`adjunct routing link`** step, the announcement must not contain any information essential to the caller, such as further instructions, since the announcement immediately terminates when the switch receives a destination from the ASAI adjunct.
  - If a **`wait-time`** step follows immediately after an **`adjunct routing link`** step, the step must specify ringback or music, but not silence, as the feedback option. If the caller hears silence, the caller might abandon the call.

  Important:

  If an ASAI link or application specified in the **`adjunct routing link`** step is out of service, the step is skipped. If the next step is not a **`wait-time`**, **`announcement`**, or **`adjunct routing link`** step, as much as six minutes can elapse before the switch determines that the adjunct application is out of service.

- The field **IC Adjunct Routing** should be disabled on the CTI Link screen unless the CTI application requires it enabled. Then, program the associated vectors as instructed for that application.
- When IC Adjunct Routing is disabled, only the first **`announcement`** or **`wait-time`** step encountered is skipped after a sequence of adjunct route steps fails. The following paragraphs apply when IC Adjunct Routing is disabled.
- The second step after the **`adjunct routing link`** step must be implemented as a default treatment in case the host application or ASAI link is down. Speed of execution for the default treatment step, for example, **`route-to number`** *0* if unconditionally, is controlled by the following factors:
  - If the ASAI link is down and if the first non-adjunct routing link step is a wait-time or an announcement treatment, the treatment step is skipped and the default step that follows the skipped, treatment executes immediately.

  - If the host application is not down, the default step executes if the adjunct does not provide a route within the time defined by the first non-adjunct step. For example, if the first non-adjunct step is an announcement, the default step executes after the time defined by the length of the announcement is exceeded.

- When a vector contains an **`adjunct routing link`** command and an ASAI link or application fails, special rules apply to vector processing operations that result. Consider the special processing operations while designing adjunct routing vectors.
- Vector processing continues while an ASAI call route request is processed at an adjunct. Succeeding vector steps can terminate an ASAI call route request if the steps execute before a call route can be provided by the adjunct. Alternately, the adjunct can reject the call route request, and subsequent vector processing proceeds normally.
- The **`wait-time hearing i-silent`** command allows the adjunct to decide whether to accept an incoming ISDN-PRI call. When this step is encountered after an adjunct routing link step, the switch does not return an ISDN PROGRESS message to the originating switch. The **`wait-time hearing i-silent`** command is important for the Network ISDN and LAI features.

Receiving and implementing an ASAI call route

Communication Manager that receives an adjunct-supplied call route performs various checks to validate the call route implementation. When the adjunct-supplied route is validated, the resulting operations are similar to those in effect for a **`route-to`** *xxxxx* **`with coverage = y`** command. The caller hears normal call progress tones and feedback, and if the call routes to an extension with no available call appearances and no coverage path, the caller hears a busy signal.

Other features in effect at the adjunct-supplied destination, such as Send-All-Calls or Call Forwarding, interact with the routed call.

Look-Ahead Interflow operations are not applied when calls are routed over trunks. Instead, ASAI-routed calls are directed to their adjunct-supplied destination without waiting for call acceptance.

Validation for an adjunct-supplied call route

When the switch receives adjunct-supplied call route instructions, the switch validates the route according to the following process:

1. The switch verifies that the COR rules specified for the target VDN permit the call to be terminated at the adjunct-supplied destination.
2. The switch validates the following information:
   - Destination number
   - ACD split
   - TAC/AAR/ARS access code
   - Dial plan compatibility
   - Other options specified by the adjunct

3. If the ASAI adjunct specifies the direct agent call option, the destination number (agent) must be logged into the adjunct-specified ACD split.
4. If the destination for the call is external, the switch verifies that a trunk is available for the call.

Switch response to validated adjunct-supplied call routes

If the switch validates an adjunct-supplied call route, the following operations occur:

1. Vector processing in the VDN that contains the initiating **`adjunct routing link`** command terminates immediately.
2. The switch signals the ASAI adjunct that the route is accepted.
3. The switch routes the call to the destination specified by the ASAI adjunct.

Switch response to invalid adjunct-supplied call routes

If any of requirements for call route validation listed in Validation requirements for an adjunct-supplied call route are not met, items the following operations occur:

1. The switch discards the route.
2. The switch signals the ASAI adjunct that the route is invalid.
3. Vector processing of any other default treatment steps in the VDN that contains the initiating adjunct routing link proceeds.

Data sent with an ASAI call route request

When a call encounters an **`adjunct routing link`** command and if the call is not queued to a split, the switch sends an ASAI message that requests a call route over the specified adjunct link. The following list identifies the contents of the message, along with a comment or a brief explanation for each item:

Calling number information

The calling party number or billing number (CPN/BN) that is provided by ISDN-PRI or R2-MFC signaling facilities. If the call originates from a local switch extension, this extension is the calling number.

Originating line information (II-digits)

A two-digit code that is provided by ISDN-PRI facilities that indicates the type of originating line. This information is not provided by SIP facilities.

Called number

The originally called extension if a call is forwarded to a VDN, or the first VDN through which the call was routed if the call was not forwarded to the VDN.

If the VDN Override for the Trunk ASAI Messages feature is in effect for an incoming call, the active VDN extension (instead of the Called Number received in the SETUP or INVITE message) is sent in the Called Number for the Call Offered, Alerting, Queued, Connect, and Adjunct Route-Request ASAI Event Reports.

Routing VDN

The last VDN that routed the call to the vector that contains the **`adjunct routing link`** command.

Call identifier

An ASAI identifier that permits the ASAI adjunct to track multiple calls by either Event Notification or 3rd Party Call Control. For more information on ASAI, see *Communication Manager CallVisor ASAI Technical Reference*.

Enhanced Information Forwarding (related data) and Look-Ahead Interflow information (if any)

Includes the original VDN display information, the priority level of the call at the originating switch, and the time that the call entered vector processing.

Digits collected by Call Prompting or Caller Information Forwarding (CINFO) (if any, maximum of 16 digits)

Digits that are collected by the most recent **`collect digits`** command. For more information, see "Call Prompting", "ANI /II-digits routing" and "Caller Information Forwarding (CINFO)" in the *Programming Call Vectoring Features in Avaya Aura® Call Center Elite* document.

User-to-User Information (UUI)

User-provided data that is associated with the call. If provided by ASAI, this data was provided in a 3rd-Party-Make-Call, Auto-Dial, or Route-Select message. If provided over an ISDN or SIP trunk, the data was in the SETUP or INVITE message that delivered the call to this switch. Calls that contain UUI specifically used by ASAI allow ASAI UUI to be propagated to the new call during a manual transfer or conference operation. ASAI UUI is propagated to a new call during its establishment when the agent presses the transfer/conference button the first time. If the call is transferred to a remote switch, the ASAI UUI from the first call is copied into the SETUP or INVITE message sent for the second call, in which case, the alerting event message sent to an ASAI application contains the ASAI information.

Special vector processing considerations associated with adjunct routing

When you design call vectors that include more than one **`adjunct routing link`** command, you must be aware of a number of special operational features.

Effects of ASAI link/application failure on vector processing

An ASAI link failure can change the manner in which subsequent announcement or wait-time treatment steps are processed.

In the following simplified vector example, the step that follows immediately after an **`adjunct routing link`** command is a **`wait-time`** command. If the adjunct routing link step fails at either the ASAI link or adjunct application, the wait-time step is skipped.

The second step after the adjunct routing link step is often implemented as a default treatment. In the example shown above, the default treatment in step 3 is a route to an attendant. If the switch recognizes that the ASAI link or adjunct application is out of service, this step executes immediately. Otherwise, the step executes only if the application does not respond with a route within 60 seconds (the wait-time assigned in the example).

Simplified example of vector processing in an ASAI link/application failure condition

```
1. adjunct routing link 11  [link/application is down]
2. wait-time 60 seconds hearing ringback  [step is skipped]
3. route-to number 0 with cov n if unconditionally [step is executed]
4. disconnect after announcement 2000
```

Vector processing with goto steps in an ASAI link or application failure condition

Processing rules for a vector that includes more than one **adjunct routing link** command and has an ASAI link or application failure condition in effect summarized as follows:

- An announcement or wait time treatment is skipped when one of the following conditions is true:
  - The treatment step follows immediately after a failed **adjunct routing link** command.
  - The treatment step is the first non-goto step that follows a **goto** step that succeeds. In this context, a **goto** step is treated as a success when the specified goto condition is true and the call branches from the **goto** step to the treatment step.
  - The treatment step is the first non-goto step that follows a failed **goto** step. In this context, a **goto** step is treated as a failure when the specified **goto** condition is true, the call fails to branch and control proceeds to the treatment, which is the next step listed in the vector sequence.

Note:

The treatment step is skipped even when a failed **goto** step that precedes the treatment is, in turn, preceded by more than one successful **goto** steps.

The rules listed for vector processing under ASAI link or application failure conditions are illustrated in the following examples.

Example 1 - Vector processing with goto steps in an ASAI link or application failure condition

```
VDN (extension=1040  name=‘‘Ad Route’’  vector=40)
Vector 40
 1. adjunct routing link 10  [link/application is down]
 2. wait-time 10 seconds hearing ringback [step is skipped]
 3. adjunct routing link 20 [link/application is down]
 4. goto step 7 if available-agents in split 20 < 1 [step executes and condition is
false]
 5. wait-time 10 seconds hearing ringback [step is skipped]
 6. goto vector 50 @step 1 if unconditionally  [step executes, go to vector 50]
 7. goto step 10 if calls-queued in split 20 pri l > 50
 8. announcement 4001
 9. goto vector 50 @step 1 if unconditionally
10. route-to number 6000 with cov n if unconditionally
   VDN (extension=6000  name=‘‘Message’’  vector=60)
```

Based on the scenario presented in the example shown above, the following vector processing events occur:

Step 1 fails: In the example, the adjunct link or application is out of service. The **`adjunct routing link`** command in step 1 fails.

Step 2 is skipped: Because the **`wait-time`** command in step 2 immediately follows an **`adjunct routing link`** command whose adjunct link is out of service, the **`wait-time`** step is skipped.

Step 3 fails: For purposes of this example, step 3 contains another **`adjunct routing link`** command whose adjunct link is out of service. The step fails and control is passed to the **`goto step`** command in step 4.

Step 4 executes: A **`goto`** step that immediately follows a failed **`adjunct routing link`** command is always executed. In this example, the command fails to branch because there is a minimum of one available agent in split 20.

Step 5 is skipped: The **`wait-time`** step that follows the unsuccessful **`goto`** step (step 4) is skipped, because in an ASAI link failure condition, the first non-goto step to be processed after the first successful first **`goto`** step is always skipped if the step is either an **`announcement`** or a **`wait-time`** step. Control is passed to the **`goto vector`** command in step 6.

Step 6 executes: Step 6 routes the call to vector 50 (not shown), which is designed to queue the call and provide standard call treatment.

In the next example, the **`goto`** step succeeds when the specified condition is true, that is, no agents are available in split 20, and control is passed to step 7, where another **`goto`** step determines whether there are more than 50 calls in split 20. If the condition is true, step 7 succeeds and control is sent to step 10, where the **`route-to number`** command sends the call to vector 60.

The example processing events are described as follows:

Example 2 - Vector processing with goto steps in an ASAI link or application failure condition

```
VDN (extension=1040  name=‘‘Ad Route’’  vector=40)
Vector 40
 1. adjunct routing link 10 [link/application is down]
 2. wait-time 10 seconds hearing ringback [step is skipped]
 3. adjunct routing link 20 [link/application is down]
 4. goto step 7 if available-agents in split 20 < 1 [step executes and condition is
true]
 5. wait-time 10 seconds hearing ringback
 6. goto vector 50 if unconditionally
 7. goto step 10 if calls-queued in split 20 pri l > 50 [step executes and condition is
true]
 8. announcement 4001
 9. goto vector 50 if unconditionally
10. route-to number 6000 with cov n if unconditionally [step executes unconditionally]

VDN (extension=6000  name=‘‘Message’’  vector=60)
Vector 60

1. announcement 4000 [“We’re sorry. We are still unable to connect you to an agent. If
you would like to leave a
   message, please do so after the tone.”]
2. wait-time 6 seconds hearing silence
3. messaging split 18 for extension 1500
4. announcement 4010 [“We’re sorry. We are unable to connect you to our voice mail. If
you would like to try to
   leave a message again, please do so after the tone. Otherwise, call back weekdays
```

```
between  8:00 A.M. and 5:00 P.M.”]
5. goto step 2 if unconditionally
```

Based on the scenario presented in the example, the following vector processing events occur:

Step 1 fails: For purposes of this example, the adjunct link or application is out of service. The **adjunct routing link** command in step 1 fails.

Step 2 is skipped: Because the **wait-time** command in step 2 immediately follows an **adjunct routing link** command whose adjunct link is out of service, the **wait-time** step is skipped.

Step 3 fails: For purposes of this example, step 3 contains another **adjunct routing link** command whose adjunct link or application is also out of service. The step fails and control is passed to the **goto step** command in step 4.

Step 4 executes: A **goto** step that follows a failed **adjunct routing link** command is always executed. In this example, the command succeeds and branches to step 7, because no agents are available in split 20.

Step 7 executes: Again, a **goto** step that follows a failed **adjunct routing link** command is always executed. In this example, the command branches unconditionally to vector 60

Step 10 executes: In this example, step 10 (route-to number) is the first non-goto step immediately preceded by more than one **goto** step in an ASAI link fail condition. The step executes, because the step is not an **announcement** or **wait time** command.

Vector 60: Step 1 executes: The first step in this vector is an **announcement** command. In this example, this is the first step in the processing sequence to be either an **announcement** or **wait-time** step. However, this step is not skipped, since the step is not the first non-go to step in the processing sequence. Instead, step 10 in vector 40 (a route-to number step) is the first non-goto step.

Simultaneous processing of vector steps and ASAI call route requests

When Communication Manager sends a route request to an ASAI adjunct, vector processing continues for any vector steps that follow the **adjunct routing link** command. Therefore, non-adjunct routing link step that follows immediately after an adjunct routing link step or multiple adjunct routing link steps in uninterrupted succession can determine:

- The maximum length of time that Communication Manager waits for a call route reply from the ASAI adjunct.
- In some cases, whether or not the ASAI call route request is allowed to finish processing.

If the next step is not a **wait-time**, **announcement**, or another **adjunct routing link** command, as much as six minutes can elapse before Communication Manager determines that the adjunct application is out of service. For this reason, design vectors so that the next step to follow an **adjunct routing link** command is either a **wait-time**, or **announcement** command.

Vector steps that terminate an ASAI call route request

If an adjunct routing link step is followed by a wait-time or announcement treatment and the treatment completes before an ASAI call route request is returned by the adjunct, call processing continues for any vector steps that follow the treatment. In this case, certain vector commands terminate the ASAI call route request after being executed. Vector commands that terminate an active ASAI call route request include:

- **`busy`**
- **`check split`**
- **`converse-on split`**
- **`queue-to split`**
- **`collect digits`**
- **`disconnect`**
- **`messaging split`**
- **`route-to`**

If Communication Manager receives a valid ASAI call route message before one of the listed vector commands can execute, Communication Manager routes the call to the destination specified by the adjunct route. Otherwise, the ASAI route request is terminated.

Note:

The adjunct can also reject a call request by negatively acknowledging the route request sent by Communication Manager. When Communication Manager receives a route request rejection message from the adjunct, Communication Manager terminates any announcement or wait-time step being executed. Call processing then continues with the next vector step.

Adjunct routing-initiated path replacement

You can use QSIG or DCS with Reroute using ISDN SSE for path replacement for calls in a queue. For calls that are waiting in a queue or in vector processing, even if the call is not connected to an answering user, Communication Manager can attempt a path replacement to find the optimum path for the call.

When adjunct routing is used with a call, path replacement can be initiated when the following criteria are true:

- The inbound call is over a QSIG trunk or DCS SSE trunk.
- A route select response is received from the CTI application after the **`adjunct route`** vector command has been executed.
- The routing destination that is contained in the route select ASAI message is to an outbound QSIG trunk or out bound DCS SSE trunk.

When all three criteria are met, the trunk is seized and used for the call.

Example vector for adjunct routing-implemented path replacement

The following Call Vector example shows how a vector for adjunct routing can be written to trigger path-replacement at the terminating switch.

Note:

In order for a path-replacement to be attempted, the incoming and outgoing trunks that are used for the call must be administered with the **Supplementary Service Protocol** field set to `b`.

Adjunct routing-initiated path-replacement vector

```
1. announcement 5996
2. adjunct routing link 11
3. wait 20 seconds hearing ringback
4. announcement 3111
```

At the terminating (receiving) switch, the vector that is executed by the incoming call must be programmed with an **`announcement`**, **`wait hearing music`**, or **`wait hearing ringback`** vector command. The use of one of these commands is what makes it possible for path-replacement to take place while the call is in vector processing.

Phantom calls

A phantom call originates from a nonphysical device by way of an ASAI application and can be placed anywhere. In general, phantom calls

- Use less resources.
- Are treated as voice calls.

How do phantom calls work?

First, an application requests a phantom call by sending an ASAI third_party_make_call or auto_dial capability message to Communication Manager.

If the specific extension of a station Administration Without Hardware (AWOH) is specified as the originator, Communication Manager places the call from that extension if the extension is available.

You can specify a hunt group extension with members that are AWOH extensions as the originator.

How are phantom calls used?

Applications use phantom calls to originate a call without using a physical device. For example, applications require to:

Action	Description
Reserve a queue slot	Many call centers handle incoming requests as voice, video, data, voice messages, faxes, and e-mail. Agents handle the mix of requests. However, a single queue must manage and distribute the work load for the agents. For each non-voice request, the application can place a phantom call into the queue. When the phantom call reaches the head of the queue, the call is delivered to the agent. The agent is then given the corresponding work item on the desktop, for example, the fax.

Table continues…

Action	Description
Conference control	Multiple parties, both internal and external, can be conferenced into a call. The initial call is placed as a phantom call. When answered, the call is placed on hold by the application and another phantom call is made. The two calls are then conferenced together. This process is repeated until all parties are added to the call.
Help with trunk-to-trunk transfers	Working with the Single Step Conference (SSC) feature, applications can use the phantom call feature to help with trunk-to-trunk transfers, that is, transferring a trunk-to-trunk call to another trunk.
Alerts (wake-up, maintenance, and security)	Applications can use phantom calls to alert users of various conditions such as wake-up, maintenance, or security.

How do phantom calls affect Call Vectoring?

Because phantom calls can be directed anywhere, you must properly configure the application and Communication Manager to ensure that the vector commands that are executed for the calls make sense. For more information, see *Communication Manager CallVisor ASAI Technical Reference*.

Communication Manager does not block phantom calls from executing any vector commands because phantom calls follow the same vector processing as regular voice calls. However, do not allow phantom calls to enter certain vector steps such as the following.

Steps	Description
Announcements	Because there is nobody listening to an announcement that is made to a phantom call, there is no sense in playing one.
Collect steps	In a phantom call, the **`collect`** step fails because the step cannot connect a tone receiver to a station Administration Without Hardware (AWOH). The step times out because there is nobody to enter the expected digits. The **`busy`** step provides a busy signal to the caller. In a phantom call, the **`busy`** step disconnects the call because Communication Manager clears a phantom call when the call cannot terminate at a specific local destination.

Phantom call administration

There are no administration screens that are specific to phantom calls, but the following criteria must be met in order for the feature to work:

- Some stations AWOH must be administered.
- If a hunt group is specified as originator, a non-ACD hunt group with AWOH members must also be administered.
- It is recommended that meaningful names are assigned for the stations AWOH that are used by phantom calls if the calling party name will appear on the agent's or Service Observer's display.

Single-step conference

With Single-Step Conference (SSC), an application can:

- Add a device into an existing call, for example, to play announcements or make voice recordings.
- Facilitate application-initiated transfers and conferences.

Stations that are AWOH can use SSC. The party can be added to a call in the listen only mode with no visibility or in the listen/talk mode with visibility.

SSC is only available through an ASAI link.

How does SSC work with Call Vectoring?

The call to which an extension is to be single-step conferenced is not allowed to be in vector processing unless the visibility option with the single-step conference request indicates no visibility.

To be transferred to a VDN, a conference call must not have more than two parties.

Note:

Invisible (listen-only) single-step-conference parties are not counted in the two-party limit for a conference call transfer to a VDN.

Multiple outstanding route requests

This feature allows multiple ASAI route requests for the same call to be simultaneously active. The route requests can be over the same or over different ASAI links.

Route requests are all made from the same vector. The requests must be specified without intermediate **`wait-time`**, **`announcement`**, **`goto`**, or **`stop`** steps. If the **`adjunct routing link`** commands are not specified back-to-back, standard adjunct routing functionality applies and previous outstanding route requests are cancelled when an **`adjunct routing link`** vector step is executed.

The first route select response that is received by Communication Manager is used as the route for the call and all other outstanding route requests for the call are canceled.

With multiple outstanding route requests, multiple adjuncts can process the route call request without waiting for the first route attempt to fail. An application can make use of this feature to distribute the incoming call load evenly across adjuncts based on the adjunct's current CPU load.

Note:

Each link has a unique extension number, even in a configuration where there can be multiple links to the same adjunct.

Multiple call route request example

The following example shows a typical vector where multiple adjunct route requests to multiple links are active at the same time. The first adjunct to route the call is the active adjunct and specifies which VDN the call must be routed to at that point.

Sample adjunct routing link vector with redundancy

```
1. wait-time 0 seconds hearing ringback
2. adjunct routing link 11
3. adjunct routing link 12
4. adjunct routing link 13
5. wait-time 6 seconds hearing ringback
6. route-to number 1847 with cov n if unconditionally
```

Advanced Vector Routing features

You can use the following advanced vector routing features to enhance the conditional routing capabilities of basic call vectoring, and to improve the operational efficiency of a call center.

Rolling Average Speed of Answer (ASA)

You can use Rolling ASA Routing to make routing decisions based on the current average time a skill or VDN takes to answer a call. This ensures that the vectors route calls to the skill or to the VDN with the shortest average time.

Expected Wait Time (EWT)

You can use EWT Routing to make routing decisions based on the wait time for a call or split, in queue. EWT can be passed to an Interactive Voice Response (IVR) or a Voice Response Unit (VRU) to notify the caller about the expected time in queue.

VDN Calls

You can use VDN Calls Routing to make routing decisions based on the number of incoming trunk calls currently active in a VDN. With the VDN calls conditional, you can use a vector to limit the number of simultaneous calls to a particular VDN. For example, if you enter into a contract with a service agency to handle 100 simultaneous calls for a client, you can route calls in excess of 100 to a **busy** step.

Advanced Vector Routing command set

The commands used in Advanced Vector Routing are listed in the following table.

Command category	Action taken	Command
Routing		
	Queue the call to a backup ACD split.	**check split**
Branching/programming		
	Go to a vector step. Go to another vector.	**goto step** **goto vector**

When to use wait time predictions

A number of factors can affect the accuracy of wait time predictions. Wait time predictions are best suited for medium-volume or high-volume call scenarios. The potential accuracy of a wait time predictor increases as the rate of removal from queue increases.

Under all conditions, EWT is the most accurate wait time predictor, but EWT is most accurate when the rate of removal from queue at a given split priority level is a minimum of one call every 30 seconds.

Predictions can be made for a split with multiple priority levels as long as the majority of calls are delivered to lower priority levels. If the majority of calls are queued at the higher-priority levels, any predictions made for the lower-priority levels not be accurate.

The following circumstances can limit the accuracy of the wait time predictions.

Factors limiting wait time prediction accuracy	Description
System restart or new split administration	The EWT algorithm uses a combination of historical and real-time information to make predictions. When no historical information exists, such as when a new split is added or a reset system 3 or 4 is completed, there is the potential for inaccuracies. To prevent inaccurate predictions when there is no historical information, administer the **Expected Call Handling Time** field on the Hunt Group screen. The value in this field is then used in place of the missing historical data. If the value of this field does not accurately reflect the call handling times of the split, EWT predictions can be inaccurate until some call history is generated. The algorithm normally requires about 30 queued calls to be answered from a split priority level before it reaches its maximum accuracy. You can change the value in the **Expected Call Handling Time** field by executing a `change hunt group` command. Changing the value does not disrupt EWT predictions by overwriting EWT history. The value is stored and used the next time a reset system 3 or 4 is executed.
Low call volume applications	Split priority levels where the rate of removal from the queue is very low can only be predicted with limited accuracy.
Sites with frequent staffing changes	Although EWT immediately adjusts for all types of staffing changes, since predictions can already be made for calls that are waiting in queue, the past predictions were based on staffing information which is now out of date. Therefore, the EWT in scenarios where large staffing changes are continually happening can only be predicted with limited accuracy.

Table continues…

Factors limiting wait time prediction accuracy	Description
Staffed agents who rarely answer calls to a split	The EWT algorithm takes account of agents in multiple splits in its calculation. However, suppose there are many agents who are assigned to a split but spend most of their time answering calls in their other splits. If a large number of these agents are moved to or from the split, the EWT for this split be temporarily inaccurate until it adjusts to those changes.
Applications with widely varying call handling times	If the majority of calls to a split are handled within a narrow range of times, the accuracy of any predictor will be much greater than that for a split where call handling times are widely different.

Expected Wait Time (EWT)

With EWT routing, you can make routing decisions based on the time that a caller can expect to wait in queue.

How EWT is calculated

Depending on how the EWT condition is used in a vector step, the predicted wait time calculation is derived by the following rules:

- If the call is currently queued to a split, the EWT is based on the actual current position of the call in the queue at a particular priority level and the rate of service of calls from the queue at that priority level.
- If the call is not yet queued to a split, the EWT is based on the assumption that the call is placed at the end of the queue.

EWT also adjusts for many other factors such as multiple split queuing, call handling times, and the impact of direct agent calls on the wait time of other calls to the split. The algorithm adjusts EWT immediately for changes in staffing, such as agents logging in or taking breaks in the AUX work mode.

The EWT can also be passed to a VRU so that a caller can be notified of the expected time in queue. The expected-wait condition can be used with either the **`goto`** or **`check`** commands.

Call Vectoring offers several conditionals to estimate predicted wait time on a queue, including EWT, rolling ASA and Oldest Call Waiting (OCW), but EWT uses the most accurate method of prediction. EWT checks the real-time and historical information, such as priority level, position in queue, and number of working agents.

EWT is responsive to changing call center conditions. For example, EWT adjusts instantly to any staffing changes in the split, or if agents moves in or out of auxiliary work mode, the wait time predictions immediately adjust.

EWT does not include the time in a call vector before the call enters a queue. It also does not include the time that the call rings at a telephone after it is removed from the queue.

EWT for a split

The EWT for a split is the time that an incoming call is expected to remain in queue if the call is queued to the split at the specified priority level. EWT is used to determine if a call must be queued to the split.

The following vector example shows how to use EWT to determine if a call must be queued to a split.

```
1. goto step 3 if expected-wait for split 1 pri l < 600
2. busy
3. queue-to split 1 pri l
4. announcement 3001
5. wait-time 998 secs hearing music
```

In the example shown above, the following wait time conditions are possible:

- If there are agents available, EWT is zero.
- EWT is infinite if:
  - There are no logged-in agents.
  - All logged-in agents are in the AUX work mode.
  - The split queue is full.
  - There is no split queue and all agents are busy.
  - The split queue is locked. This occurs when the last working agent in a non-vector-controlled split attempts to enter the AUX work mode.

EWT for a call

EWT for a call is the remaining time that a caller can expect to wait before the call is serviced from the queue. If the call is queued to multiple splits, the remaining queue time for each split is calculated, and the shortest calculation is used as the EWT.

A call must be queued to a minimum of one split to have an expected wait time. If the call is not queued, or if the call is queued to splits that are not staffed, the EWT value is infinite.

The following vector example shows how to use EWT to determine call treatment:

```
1. queue-to split 1 pri m
2. check split 2 pri m if expected-wait < 30
3. goto step 5 if expected-wait for call < 9999
4. busy
5. announcement 3001
6. wait-time 998 secs hearing music
```

Passing EWT to a VRU

The Expected Wait Time (EWT) for a call can be passed to a Voice Response Unit (VRU) to inform callers about the expected time in queue. EWT is passed to the VRU with the **converse-on** command as wait data. The value that is outpulsed to the VRU is the expected wait time of the call in seconds. The VRU then converts the seconds to a spoken message. The expected wait is calculated after the VRU port answers the call, so queuing to a converse split does not adversely impact the EWT value that is passed to the VRU.

No zero padding is added to the wait time that is passed to the VRU. If the EWT for the call is 128 seconds, the digits 1, 2, and 8 are outpulsed. If the EWT is 5 seconds, the digit 5 is outpulsed.

The wait time passed to the VRU is the most accurate prediction possible. On an average, 50 percent of the time the actual wait time is shorter and 50 percent of the time it is longer. Configure the VRU applications to make an upward adjustment of the prediction so that the majority of callers receive a predicted wait time that is either equal to or greater than the actual wait time.

The VRU can also announce EWT at set intervals while the call is in queue, but use this strategy with caution. Circumstances such as a reduction in the number of agents or a sudden influx of higher priority calls can cause the EWT of the caller to increase from one announcement to the next.

If the call is not queued or is queued only to splits that are unstaffed, or to splits where all agents are in the AUX work mode, the end-of-string character # is the only data item that is outpulsed to the VRU.

The following vector example illustrates routing based on the predicted split wait time and passing wait data to the VRU. The caller hears the wait time only if the caller is expected to wait for more than 60 seconds in queue. Callers who have to wait more than 10 minutes are requested to call back later.

```
1. goto step 3 if expected-wait for split 32 pri l < 600
2. disconnect after announcement 13976
3. queue-to split 32 pri l
4. wait-time 20 secs hearing ringback
5. goto step 7 if expected-wait for call < 40
6. converse-on split 80 pri l passing wait and none
7. announcement 11000
8. wait-time 60 secs hearing music
9. goto step 7 if unconditionally
```

Calls that have predicted wait times greater than 10 minutes fail step 1 and are disconnected after an announcement. If the expected wait time is less than 10 minutes step 1 routes the call to step 3 where it is queued to split 32 and waits 20 seconds hearing ringback. After 20 seconds if the expected wait time for the call is less than 40 seconds, step 5 routes the call to an announcement followed by a wait with music. If the expected wait time for the call is equal to or greater than 40 seconds, step 6 informs the caller of the amount of time the caller can expect to wait before the call is answered.

Notifying callers of wait time without a VRU

You can use EWT to notify callers of their expected wait time without a VRU. This can be done using recorded announcements and by associating each recorded announcement with a time band, as shown in the following example.

```
VECTOR 101
1. queue-to split 3 pri h
2. goto step 4 if expected-wait for call <= 600
3. busy
4. wait-time 12 seconds hearing ringback
5. announcement 3001 [Thank you for calling ABC Inc. All agents are busy, please wait
and we will get to your call as soon as possible]
6. goto vector 202 if unconditionally
_ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _
VECTOR 202
```

```
1. goto step 13 if expected-wait for call > 280
2. goto step 11 if expected-wait for call > 165
3. goto step 9 if expected-wait for call > 110
4. goto step 7 if expected-wait for call > 55
5. announcement 3501 [Thank you for waiting. Your call will be answered within the next
minute.]
6. goto step 14 if unconditionally
7. announcement 3502 [Thank you for waiting. Your call will be answered within
approximately one to two minutes.]
8. goto step 14 if unconditionally
9.  announcement 3503 [Thank you for waiting. Your call will be answered within
approximately two to three minutes.]
10. goto step 14 if unconditionally
11. announcement 3504 [Thank you for waiting. Your call will be answered within
approximately three to five minutes.]
12. goto step 14 if unconditionally
13. announcement 3505 [We apologize for the delay. Due to heavy call volume, you may
have to wait longer than five minutes
    to speak to a representative. If possible, we suggest that you call between the
hours of 8am and 10am for the
    fastest service.]
14. wait-time 120 secs hearing music
15. goto step 1 if unconditionally
```

In step 1 of the example shown above, the call is queued to split 3 at high priority. If the call fails to get a queue slot in split 3, if split 3 has no working agents, or if the wait time in split 3 at high priority exceeds 10 minutes, step 2 fails and the caller receives a busy signal. If step 2 succeeds, the caller hears ringback and an announcement and is then sent to vector 202.

Steps 1 through 4 of vector 202 determine tests to determine a predicted time band interval for the remaining queuing time for the call. One of five recorded announcements is then played to provide the caller with an expected wait time.

You can program vectors so that few callers experience wait times that exceed the wait time of the announcement. In the example shown above, the EWT thresholds are set lower than the times that are quoted in the recorded announcements.

Using EWT to route to the best split

You can use EWT to change the normal queuing strategy for multiple splits to ensure that calls are answered in the shortest possible time. However, this strategy uses additional system resources and can make reading and analyzing split reports difficult. Alternately, you can use EWT to identify the best split for each call and not use multiple split queuing.

The following vector example shows a scenario that includes a main split (1) and a backup split (2). In this example, the preference is for an agent from the main split service the call, but a 30-second maximum wait time is a competing preference.

The strategy in this vector is to use the backup split only if the backup split can answer the call within 30 seconds and the main split cannot.

```
1. goto step 5 if expected-wait for split 1 pri m <= 30
2. goto step 5 if expected-wait for split 2 pri m > 30
3. check split 2 pri m if unconditionally
4. goto step 6 if unconditionally
5. queue-to split 1 pri m
6. wait-time 12 secs hearing ringback
7. announcement 3501
8. converse-on split 18 pri m passing wait and none
```

```
9. wait-time 120 secs hearing music
10. goto step 8 if unconditionally
```

In the example, step 1 branches to step 5 if the main split can answer the call within 30 seconds. If the main split cannot answer the call within 30 seconds, step 2 checks to see if the backup split can answer the call within 30 seconds. If the test fails, the call branches to step 5 and is queued to the main split. If possible, the call is queued to the backup split in step 3. At this point, the call is queued either to the main split or to the backup split, but not to both.

Steps 6 through 10 provide audible feedback to the caller while the call is in the queue. Note that in step 8, which is executed every 2 minutes, a VRU is used to provide the caller with the remaining wait time.

Factors that increase EWT for a split priority level

The most common causes for an increase in EWT for a split priority level are:

- The number of calls that are in queue increases.
- Agents log out.
- Agents go on break or are otherwise in the AUX work mode.
- Agents are moved to another split.
- Agents with multiple splits answer an increasing number of calls in other splits.

Following are the other conditions that can increase the EWT for a split priority level:

- The average talk time increases.
- The number of calls at a higher priority increases.
- The number of direct agent calls increases.
- The number of RONA calls increases.
- The number of abandoned calls decreases.
- The number of calls that are queued in this split but answered in another decreases.

Factors that decrease EWT for a split priority level

The most common causes for a decrease in EWT for a split priority level are:

- The number of calls in queue decreases.
- Agents log in and start answering calls.
- Agents return from break or otherwise are no longer in the AUX work mode.
- Agents are moved from another split.
- Agents with multiple splits answer fewer calls in other splits.

The following conditions can reduce the EWT for a split priority level:

- The average talk time decreases.

- The number of calls at higher priority decreases.
- The number of direct agent calls decreases.
- The number of RONA calls decreases.
- The number of abandoned calls increases.
- The number of calls queued in this split but answered in another increases.

Rolling Average Speed of Answer (ASA)

Rolling ASA Routing helps you to make routing decisions that are based on the current average time that it takes for a call to be answered in a split or VDN. In this way, a vector can route a call to the VDN or split where it is likely to be answered most quickly.

Rolling ASA versus interval ASA

The ASA calculation used for vector routing is called “Rolling ASA” to differentiate it from the “Interval ASA” that is recorded in Basic Call Management System (BCMS) and Avaya Call Management System (CMS) reports.

Rolling ASA is a running calculation that does not take into account the 15-minute, half-hour, or hour reporting intervals. Rolling ASA does not reflect interval boundaries.

The Interval ASA used for BCMS and CMS reports is calculated on reporting interval boundaries and clears to zero at the start of each reporting interval.

The Rolling ASA for a split or VDN is calculated based on the speed of answer for all calls recorded since system start up and is recalculated every time a call is answered. During each calculation, the speed of answer for the current call is given a weighted value that is greater than previous calls. Approximately 95 percent of the value of Rolling ASA is retrieved from the previous ten calls.

Note:

Calls that are not answered, such as calls that receive a forced busy, are not included in the Rolling ASA calculation.

The Rolling ASA is calculated for an entire split or VDN. The calculation does not include the priority levels of answered calls.

When to use rolling ASA

Use Rolling ASA to test whether vector processing queues the call to additional splits or skills when the main split or skill does not currently meet the targeted threshold.

Do not use Rolling ASA conditionals to prevent calls from queuing to the main split or skill, or being answered in the principal VDN. If no calls are being answered in the main split or skill or VDN, the value of rolling ASA does not change. This can result in all future calls being locked out of the main split or skill, or VDN unless there are other call vectors in the system directing calls.

Important:

To implement a call flow that tests whether or not to queue a call to a main split or skill, use the EWT feature.

Rolling ASA split calculation

The rolling ASA for a split is the average call answer time, as specified by the time interval that starts when call processing attempts termination to a split, and ends when the call is answered in that split. The measured interval includes both time in queue and ringing time at the agent station.

If the call is answered in another split or the call is abandoned by the caller, rolling ASA is not recorded for the call. If a call flows into a split from another split, the time queued and ring time for the previous split are not included. If a call is queued in multiple splits, only the rolling ASA for the split in which the call is answered is measured.

Rolling ASA VDN Calculation

The rolling ASA for a VDN is the average call answer time, as specified by the time interval that starts when call processing is initiated within the VDN until it is answered. The measured interval includes:

- Time elapsed in vector processing, including time in announcements.
- If the call is answered by an agent, time in queue and time ringing at the agent station.

Note:

If a call flows between VDNs, only the time elapsed in the answering VDN is used in the calculation.

Rules for specifying VDNs

Rolling ASA follows the rules used for other Advanced Vector Routing conditionals to specify a VDN in a goto step:

- A VDN number.
- The value designated as *latest*. The latest VDN is the VDN currently processing the call. The latest VDN is not affected by VDN override settings.
- The value *active*. The active VDN is the VDN of record, which is the called VDN as modified by override rules. For example, if a call routes from a VDN with override set to *yes* then the new VDN is the active VDN. If a call routes from a VDN with override set to *no*, the previous VDN is the active VDN.

Combining VDN and ASA routing example

The following vector example combines VDN and split ASA routing.

```
1. queue-to split 10 pri h
2. goto step 6 if rolling-asa for split 10 <= 30
3. check split 11 pri h if rolling-asa <= 30
4. check split 12 pri h if rolling-asa <= 30
5. check split 13 pri h if rolling-asa <= 30
6. announcement 10000
```

```
7. wait-time 40 secs hearing music
8. goto step 3 if unconditionally
```

Step 1 queues the call to the main split. If the main split is currently answering calls within the target time of 30 seconds, step 2 bypasses all of the backup splits and goes directly to the announcement in step 6. The assumption is that the call will be handled by split 10 within the time constraints. However, if the call is not answered by the time that vector processing reaches step 8, the backup splits are checked.

If the rolling ASA for the main split is greater than 30 seconds, steps 3, 4, and 5 check the backup splits. The call is queued to any of these splits that have a rolling ASA of 30 seconds or less. If the call still is not answered by the time that vector processing reaches step 8, the backup splits are checked again.

VDN Calls

VDN Calls routing allows you use the counted-calls conditional to make routing decisions on the number of incoming trunk calls that are currently active in a VDN.

How VDN Call counts are calculated

The counted-calls conditional allows a vector to limit the number of simultaneous calls directed to a particular VDN. For example, if a service agency is contracted to handle 100 simultaneous calls for a client, calls in excess of that number can be routed to a busy step.

VDN Call counts follows the rules used for other Advanced Vector Routing conditionals to specify the VDN in a goto step:

- A VDN number.
- The value designated as latest. The latest VDN is the VDN currently processing the call. The latest VDN is not affected by VDN override settings.
- The value active. The active VDN is the VDN of record, which is the called VDN as modified by override rules. For example, if a call routes from a VDN with override set to **yes** then the new VDN is the active VDN. If a call routes from a VDN with override set to **no**, the previous VDN is the active VDN.

When Advanced Vector Routing is enabled, a count of active incoming trunk calls is kept for each VDN. The VDN counter increments each time that an incoming call is placed to the VDN and decremented each time that an incoming call is released. A call is treated as active in a VDN from the time the call routes to the VDN until all parties on the call are dropped and the call is released.

Note:

The call is counted for the originally called VDN only. When a call is routed to another VDN, the call counter for the subsequent VDN does not increment, nor does the call counter for the original VDN decrement.

The VDN Call count includes the following types of calls:

- Incoming trunk calls routed directly to the VDN.
- Incoming trunk night service calls in which the VDN is the night service destination.

- Calls that cover or forward to the VDN if it is the first VDN routed to and the call is an incoming trunk call.
- Already counted calls that are conferenced with counted or not counted calls from the same VDN.

The VDN call count does not include:

- Internal calls to the VDN.
- Calls that are transferred to the VDN, unless they are a SIP transfer and the **Suppress Target VDN counted-calls?** field is set to `n`.
- Calls that are redirected to their VDN return destination.
- Conferenced calls that were previously counted on different VDNs.

Using the counted-calls conditional example

The following vector example shows how the counted-calls conditional can be used to route calls.

Using VDN call counting to route calls

```
1. goto step 3 if counted-calls to vdn 1234 <= 100
2. busy
3. queue-to split 60 pri l
4. wait-time 20 seconds hearing ringback
5. announcement 27000
6. wait-time 60 seconds hearing music
7. goto step 5 unconditionally
```

If more than 100 calls are active in VDN 1234, the caller hears a busy signal and vector processing is terminated. If 100 or fewer calls are active, the call queues to split 60.

ANI/II-digits routing and CINFO

Use the Automatic Number Identification (ANI) and Information Indicator Digits (II-digits) Call Vectoring features to make vector routing decisions based on the caller identity and the originating line. Use Caller Information Forwarding (CINFO) to collect Caller Entered Digits (CED) and Customer Database Provided Digits (CDPD) for a call from the network.

Avaya uses the term ANI to denote both ANI and Calling Line Identification (CLID) which can be used interchangeably within Vectoring. ANI and II-digits, when provided with an incoming call to a VDN, are sent to Call Management System (CMS) when vector processing starts. ANI, II-digits, and CINFO are forwarded with interflowed calls. ANI and II-digits are also passed over the Adjunct Switch Application Interface (ASAI) in event reports.

CINFO command set

Command category	Action	Command
Branching/ Programming	Go to a vector step (ANI, II-digits) Go to a vector step that is based on ced or cdpd (CINFO digits)	`goto step`
	Go to another vector (ANI, II-digits) Go to another vector based on ced or cdpd. (CINFO digits)	`goto vector`
Information Collection	Pass ANI to a Voice Response Unit (VRU) Pass ced and cdpd to a VRU (CINFO)	`converse-on`
	Collect ced and cdpd from a network ISDN SETUP message.	`collect digits`
Routing	Route the call to a number that is programmed in the vector, based on ced or cdpd	`route-to number`
	Route the call to digits supplied by the network	`route-to digits`
	Request routing information from an ASAI that is based on ced or cdpd	`adjunct-routing`

ANI routing

ANI provides information on the caller's identity that can be used to improve call routing decisions. For example, calls from a specified customer can receive unique routing, local calls can be routed differently from long distance calls, or calls from different geographical areas can receive different routing. ANI can be compared against entries in a Vector Routing Table (VRT), and is supported with ISDN or SIP trunks.

ANI basics

Calling Party Number and Billing Number

ANI is based on the Calling Party Number (CPN), which is not always identical to the Billing Number. For example, if the call is placed by a user from Communication Manager, the CPN can be either the Communication Manager-based billing number or the station identification number.

String length

The ANI routing digit string can contain up to 16 digits. This supports international applications. However, ANI information in those countries that use the North American dial plan contains only 10 digits.

Call types that use ANI

The following call types have associated ANI values:

- Incoming ISDN (including PRI, BRI, and H.323) calls that send ANI.
- Incoming SIP calls that send SIP contact headers.

- Incoming R2-MFC signaling calls that send ANI.
- Distributed Communications Services (DCS) calls.
- Internal calls.

Note:

If ANI is not provided by the network for an incoming call, ANI is not available for vector processing on the call.

Use of wildcards

The `goto...if ANI` vector uses wildcards for either a direct comparison ('if ani = 123+') or comparison in a vector routing table ('if ani in table 12'). You can use either '+' or '?' wildcards.

- '+': Only one trailing, or starting '+' is allowed per digit string and it matches zero or more digits, or a single '#'. The wildcard '+' is not the same as the leading '+' that SIP stations can send on an ANI. The leading '+' makes `goto...if ANI` comparison fail.
- '?': As many '?' as possible digits placed anywhere in the digit string are allowed. A '?' matches exactly one digit in the location of the '?'.

Use with vector routing tables

ANI data can be tested against ANI numbers provided in vector routing tables.

EAS agent calls

When an EAS agent makes a call to a VDN, the agent login ID is used as the ANI in place of the physical terminal number.

Internal transfer to VDN

When a call is transferred internally to a VDN, the following outcomes can occur:

- If the transfer completes before the call reaches the ANI conditional, the ANI value of the originator of the call is used.
- If the transfer completes after the call reaches the ANI conditional, the ANI value of the terminal that executes the transfer is used.

Tip:

To ensure that the ANI of the originator is preserved during a transfer, add a filler step such as wait with silence, to the beginning of the vector. In this way, a transfer can be completed before the ANI conditional is encountered.

Use of ANI with Vector Routing Tables

You can test ANI against entries in a Vector Routing Table (VRT). VRT contain lists of numbers that can be used to test a `goto...if ani` command. ANI can be tested to see if it is either in or not-in the specified table. Entries in the tables can also use the "+" and "?" wildcards.

The following sample VRT includes various area codes for the state of California.

```
                  VECTOR ROUTING TABLE
  Number: 6              Name: California          Sort? n

1:   714+                           17: _______
2:   805+                           18: _______
3:   619+                           19: _______
```

```
4:  707+                                       20: _______
5:  209+                                       21: _______
6:  310+                                       22: _______
7:  213+                                       23: _______
8:  408+                                       24: _______
9:  510+                                       25: _______
10: 818+                                       26: _______
11: 909+                                       27: _______
12: 916+                                       28: _______
13: 415+                                       29: _______
```

The following vector example shows how calls can be routed based on information provided in the sample VRT.

```
1. announcement 45673
2. goto step 9 if ani = none
3. goto vector 8 if ani in table 6
4. queue-to split 5 pri l
5. wait-time 10 seconds hearing ringback
6. announcement 2771
7. wait-time 10 seconds hearing music
8. goto step 6 if unconditionally
9. route-to number 0 with cov y if unconditionally
```

In the example vector, the call is routed to an operator if no ANI is available for the call. If the first three numbers match an area code from table 6, the call is routed to vector 8. All other calls are queued.

Use of ANI without Vector Routing Tables

```
1. wait-time 4 secs hearing silence
2. goto step 13 if ani = none
3. goto step 12 if ani = 3035367326
4. goto vector 74920 if ani <= 9999999
5. goto vector 43902 if ani = 212+
6. goto vector 43902 if ani = 202+
7. wait-time 0 seconds hearing ringback
8. queue-to split 16 pri m
9. wait-time 120 seconds hearing 32567 then continue
10. announcement 32456
11. goto step 9 if unconditionally
12. route-to number 34527 with cov y if unconditionally
13. route-to number 0 with cov n if unconditionally
14. busy
```

In step 2, calls that do not have an associated ANI are routed to an operator. In step 3, calls are routed from a phone to a specified extension. In step 4, operator or attendant local calls are routed, which are calls with less than seven digits, to a different vector. In steps 5 and 6, calls are routed from area codes, that is, Numbering Plan Areas (NPAs) 212 and 202 to a different vector. Calls that are not rerouted by the previous vector steps are put in a queue.

II-digits routing

II-digits provide information on the originating line for a call. You can use the information for some of following purposes:

- Detect fraudulent orders for catalog sales, travel reservations, money transfers, and traveler's checks.

- Assign priority or special treatment to calls that are placed from pay phones, cellular phones, and motel phones. For example, an automobile emergency road service can prioritize calls placed from pay phones.
- Detect calls placed from pay phones when the caller intentionally tries to prevent being tracked by collection agencies or dispatching services.
- Convey the type of originating line on the agent display by routing different type calls to different VDNs.

II-digits basics

String description	II-digits is a 2-digit string that ISDN PRI provides for an incoming call. II-digits delivery is a widely available ISDN PRI AT&T Network service. This service is bundled with ANI delivery and tariffed under the MEGACOM 800® and MultiQuest 800® INFO-2 features to provide information on call origination. R2-MFC Call Category digits, when available, are treated as II-digits for routing. Leading zeros are significant. For example, the II-digits value 02 that is associated with a call does not match the digit string 2 in a vector step.
Use with a VRT	As is true for ANI routing and collected-digit routing, II-routing digits can be compared against entries in a Vector Routing Table.
Use of wildcards	The II-digits string used in a vector step or a vector routing table can contain either the + or ? wildcard.
VDN Return Destination preservation	When a call is returned to vector processing as a result of the VDN Return Destination feature, the II-digits are preserved.
Call types associated with II-digits	The following calls have associated II-digits values: • Incoming ISDN PRI calls that include II-digits. • Incoming ISDN PRI Tie Trunk DCS or non-DCS calls that include II-digits. **Note:** Since tandeming of II-digits is only supported if the trunk facilities used are ISDN PRI, traditional DCS does not support II-digits transport but DCS Plus, that is, DCS over PRI, supports II-digits transport.
Internal transfer to a VDN	When a call with II-digits is transferred internally to a VDN, the following outcomes can occur: • If the transfer completes before the call reaches the II-digits conditional, the II-digits value of the originator of the call is used. • If the transfer completes after the call reaches the II-digits conditional, the II-digits value of the terminal executing the transfer is used. Under normal circumstances, there are no II-digits for a terminal that executes a transfer. **Tip:** To ensure that the originator II-digits is preserved, add a filler step such as `wait with silence` to the beginning of the vector. In this way, a transfer can be completed before the II-digits conditional is encountered.

II-digits codes

Note:

The North American Numbering Plan Administration (NANPA) maintains the II-digit assignments. Visit http://www.nanpa.com/number_resource_info/ani_ii_assignments.html to view the latest II-digit assignments and descriptions.

II-digits assignments	
II-digits	**Description**
00	Plain Old Telephone Service (POTS) - non-coin service requiring no special treatment
01	Multiparty line (more than 2) - ANI cannot be provided on 4 or 8 party lines. The presence of this 01 code causes an Operator Number Identification (ONI) function to be performed at the distant location. The ONI feature routes the call to a CAMA operator or to an Operator Services System (OSS) for determination of the calling number.
02	ANI Failure - the originating switching system indicates (by the 02 code), to the receiving office that the calling station has not been identified. If the receiving switching system routes the call to a CAMA or OSS, the calling number can be verbally obtained and manually recorded. If manual operator identification is not available, the receiving switching system, for example, an interLATA carrier without operator capabilities, can reject the call.
03-05	Unassigned
06	Station Level Rating - The 06 digit pair is used when the customer has subscribed to a COS in order to be provided with real time billing information. For example, hotel or motels, served by PBXs, receive detailed billing information, including the calling party’s room number. When the originating switching system does not receive the detailed billing information, for example, room number, this 06 code allows the call to be routed to an operator or operator services system to obtain complete billing information. The rating or billing information is then provided to the service subscriber. This code is used only when the Directory Number (DN) is not accompanied by an automatic room or account identification.
07	Special Operator Handling Required - calls generated from stations that require further operator or OSS screening are accompanied by the 07 code. The code is used to route the call to an operator or OSS for further screening and to determine if the station has a denied-originating COS or special routing or billing procedures. If the call is unauthorized, the calling party is routed to a standard intercept message.
08-09	Unassigned
10	Not assignable - conflict with 10X test code
11	Unassigned
12-19	Not assignable - conflict with international outpulsing code
20	Automatic Identified Outward Dialing (AIOD) - without AIOD, the billing number for a PBX is the same as the PBX Directory Number (DN). With the AIOD feature, the originating line number within the PBX is provided for charging purposes. If the AIOD number is available when ANI is transmitted, code 00 is sent. If not, the PBX DN is sent with ANI code 20. In either case, the AIOD number is included in the AMA record.
21-22	Unassigned

Table continues…

II-digits assignments	
II-digits	**Description**
23	Coin or Non-Coin - on calls using database access, for example 800, ANI II 23 is used to indicate that the coin or non-coin status of the originating line cannot be positively distinguished for ANI purposes by the SSP. The ANI II pair 23 is substituted for the II pairs which otherwise indicates that the non-coin status is known, that is 00, or when there is ANI failure. ANI II 23 can be substituted for a valid 2-digit ANI pair on 0-800 calls. In all other cases, ANI II 23 must not be substituted for a valid 2-digit ANI II pair which is forward to an SSP from an EAEO. Some of the situations in which the ANI II 23 can be sent: • Calls from non-conforming end offices (CAMA or LAMA types) with combined coin/non-coin trunk groups. • 0-800 Calls • Type 1 Cellular Calls • Calls from PBX Trunks • Calls from Centrex Tie Lines
24	Code 24 identifies a toll free service call that has been translated to a POTS routable number using the toll free database that originated for any non-pay station. If the received toll free number is not converted to a POTS number, the database returns the received ANI code along with the received toll free number. Code 24 indicates a toll free service call since that fact can no longer be recognized simply by examining the called address.
25	Code 25 identifies a toll free service call that has been translated to a POTS routable number using the toll free database that originated from any pay station, including inmate telephone service. Specifically, ANI II digits 27, 29, and 70 are replaced with Code 25 under the stated condition.
26	Unassigned
27	Code 27 identifies a line connected to a pay station which uses network provided coin control signaling. II 27 is used to identify this type of pay station line irrespective of whether the pay station is provided by a LEC or a non-LEC. II 27 is transmitted from the originating end office on all calls made from these lines.
28	Unassigned
29	Prison or Inmate Service - the ANI II digit pair 29 is used to designate lines within a confinement or detention facility that are intended for inmate or detainee use and require outward call screening and restriction, for example, 0+ collect only service. A confinement or detention facility can be defined as including, but not limited to, Federal, State or Local prisons, juvenile facilities, immigration and naturalization confinement or detention facilities, which are under the administration of Federal, State, City, County, or other Governmental agencies. Prison or Inmate Service lines are identified by the customer requesting such call screening and restriction. In those cases where private pay stations are located in confinement or detention facilities and the same call restrictions applicable to Prison or Inmate Service required, the ANI II digit for Prison or Inmate Service applies if the line is identified for Prison or Inmate Service by the customer.

Table continues…

II-digits assignments	
II-digits	**Description**
30-32	Intercept - where the capability is provide to route intercept calls, either directly or after an announcement recycle, to an access tandem with an associated Telco Operator Services System, the following ANI codes must be used: • 30 - Intercept (blank) - for calls to unassigned DN. • 31 - Intercept (trouble) - for calls to DNs that have been manually placed in trouble-busy state by Telco personnel. • 32 - Intercept (regular) - for calls to recently changed or disconnected numbers.
33	Unassigned
34	Telco Operator Handled Call - after the Telco Operator Services System has handled a call for an IC, it can change the standard ANI digits to 34, before outpulsing the sequence to the IC, when the Telco performs all call handling functions, for example, billing. The code tells the IC that the BOC has performed billing on the call and the IC only has to complete the call.
35-39	Unassigned
40-49	Unrestricted Use - locally determined by carrier
50-51	Unassigned
52	Outward Wide Area Telecommunications Service (OUTWATS) - this service allows customers to make calls to a certain zone(s) or band(s) on a direct dialed basis for a flat monthly charge or for a charge based on accumulated usage. OUTWATS lines can dial station-to-station calls directly to points within the selected band(s) or zone(s). The LEC performs a screening function to determine the correct charging and routing for OUTWATS calls based on the customer's COS and the service area of the call party. When these calls are routed to the inter-exchange carrier using a combined WATS-POTS trunk group, identify the WATS calls with the ANI code 52.
53-59	Unassigned
60	TRS - ANI II digit pair 60 indicates that the associated call is a TRS call delivered to a transport carrier from a TRS provider and that the call originated from an unrestricted line, that is, a line for which there are no billing restrictions. Accordingly, if no request for alternate billing is made, the call is billed to the calling line.
61	Cellular or Wireless PCS (Type 1) - The 61 digit pair is to be forwarded to the inter-exchange carrier by the local exchange carrier for traffic originating from a cellular or wireless PCS carrier over type 1 trunks. Note: ANI information accompanying digit pair 61 identifies only the originating cellular or wireless PCS system, not the mobile directory placing the call.
62	Cellular or Wireless PCS (Type 2) - The 62 digit pair is to be forwarded to the inter-exchange carrier by the cellular or wireless PCS carrier when routing traffic over type 2 trunks through the local exchange carrier access tandem for delivery to the inter-exchange carrier. Note: ANI information accompanying digit pair 62 identifies the mobile DN placing the call but does not necessarily identify the true call point of origin.

Table continues…

II-digits assignments	
II-digits	**Description**
63	Cellular or Wireless PCS (Roaming) - The 63 digit pair is to be forwarded to the inter-exchange carrier by the cellular or wireless PCS subscriber roaming in another cellular or wireless PCS network, over type 2 trunks through the local exchange carrier access tandem for delivery to the inter-exchange carrier. Note: Use of 63 signifies that the called number is used only for network routing and must not be disclosed to the cellular or wireless PCS subscriber. Also, ANI information accompanying digit pair 63 identifies the mobile DN forwarding the call but does not necessarily identify the true forwarded-call point of origin.
64-65	Unassigned
66	TRS - ANI II digit pair 66 indicates that the associated call is a TRS call delivered to a transport carrier from a TRS provider and that the call originates from a hotel or motel. The transport carrier can use this indication, along with other information, for example, whether the call was dialed “1+” or “0+”, to determine the appropriate billing arrangement.
67	TRS - ANI II digit pair 67 indicates that the associated call is a TRS call delivered to a transport carrier from a TRS provider and that the call originated from a restricted line. Accordingly, sent paid calls must not be allowed and additional screening, if available, must be performed to determine the specific restrictions and type of alternate billing permitted.
68-69	Unassigned
70	Code 70 identifies a line connected to a pay station, including both coin and coin-less stations, which does not use network provided coin control signaling. II 70 is used to identify this type pay station line irrespective of whether the pay station is provided by a LEC or a non-LEC. II 70 is transmitted from the originating end office on all calls made from the lines.
71-79	Unassigned
80-89	Reserved for future expansion to a 3-digit code
90-92	Unassigned
93	Access for private virtual network types of service: the ANI code 93 indicates, to the IC, that the originating call is a private virtual network type of service call.
94	Unassigned
95	Unassigned - conflict with test codes 958 and 959
96-99	Unassigned

II-digits routing example

The following vector example illustrates the branching of calls that use II-digits to route to different VDNs. In this example, VDN Override is set to `yes`. In this way, the VDN name or VDN of Origin Announcement (VOA) can be used to convey to the agent the type of II-digits that are associated with the call.

```
1. goto step 9 if ii-digits = none
2. goto step 10 if ii-digits = 00
3. goto step 11 if ii-digits = 01
4. goto step 12 if ii-digits = 06
5. goto step 13 if ii-digits = 07
6. goto step 13 if ii-digits = 29
7. goto step 14 if ii-digits = 27
8. goto step 15 if ii-digits = 61
9. route-to number 1232 with cov n if unconditionally
```

```
10. route-to number 1246 with cov n if unconditionally
11. route-to number 1267 with cov n if unconditionally
12. route-to number 1298 with cov n if unconditionally
13. route-to number 1255 with cov n if unconditionally
14. route-to number 1298 with cov n if unconditionally
15. route-to number 1254 with cov n if unconditionally
```

In the example, if the call has no II-digits, step 1 branches to step 9, which routes the call to extension 1232. If the call has II-digits, steps 2 through 8 are used to route calls with different II-digits to various extensions.

CINFO

Use Caller Information Forwarding (CINFO) to associate ceds and cdpds with several vector commands to improve call processing.

The network-provided ISDN PRI SETUP message for a call includes ced and cdpd data when both of the following conditions are met:

- The incoming trunk is enabled for ISDN-PRI.
- The network uses AT&T Network Intelligent Call Processing (ICP) service.

CINFO basics

UEC IE storage

When an ISDN call is received from either the AT&T network or a tandemed PRI call, Communication Manager stores the codeset 6 User Entered Code Information Element (UEC IE) when it contains the ced or cdpd. If more than one ced UEC IE is received, only the first one is stored or tandemed with the call. If more than one cdpd UEC IE is received, only the first one is stored or tandemed with the call.

Use with collect digits commands

When a collect ced digits or collect cdpd digits step is processed, Communication Manager retrieves the ced or cdpd and places the digits in the collected digits buffer. Any digits that were previously in the collected digits buffer, such as dial-ahead digits, are erased. If a TTR was connected to the call from a previous collect digits step, the TTR is disconnected.

Valid digits are 0 through 9, *, and #. If the ced or cdpd contain invalid digits, Communication Manager does not store the UEC IE. When the collect digits step is reached, the collected digits buffer is still cleared and if a TTR is attached, it is still disconnected. A vector event is generated to indicate that no digits were collected.

If no ced or cdpd are received from the network when a collect digits step is processed, the step is not processed. However, the collected digits buffer is still cleared and if a TTR is attached, it is still disconnected.

Use of wildcards

If an asterisk (*) is included in the collected digits, it is treated as a delete character. Only the digits to the right of the asterisk are collected. If a pound sign (#) is included in the collected digits it is treated as a terminating character. Only the pound sign and the digits to the left of it are collected. If a single pound sign is sent, it is placed in the collected digits buffer.

String length

The number of ced or cdpd to collect cannot be specified in the collect digits step. Although ced and cdpb can each contain as much as 30 digits, only 16 digits can be collected and stored. If there are more than 16 digits, a vector event is generated.

Vector commands that use ced and cdpd

The following vector steps can access CINFO ced and cdpd in the collected digits buffer:

- adjunct routing link (digits passed in an event report as collected digits)
- converse-on...passing digits
- goto...if digits...
- goto...if digits in table...
- route-to digits
- route-to number...if digit...

Tip:

You can use the **callr-info** button on the phone to view the ced and cdpd information just like the other collected digits.

Internal transfer to a VDN

When a call is transferred internally to a VDN, the following outcomes can occur:

- If the transfer completes before the call reaches the CINFO conditional, the CINFO value of the originator of the call is used.
- If the transfer completes after the call reaches the CINFO conditional, the CINFO value of the terminal that executes the transfer is used.

Tip:

To ensure that the originator's CINFO is preserved during a transfer, add a filler step such as **`wait with silence`** to the beginning of the vector. In this way, a transfer can be completed before the CINFO conditional is encountered.

Buffer storage considerations

To retrieve both the ced and cdpd for a call, you must use two **`collect digits`** steps. Because the **`collect digits`** command for ced or cdpd clears the collected digits buffer, the ced or cdpd that is collected first must be used before the second set is requested.

CINFO vector example

The following vector example involves a scenario in which an incoming call enters a network enabled for the ICP service. The network Communication Manager requests information from the caller (ced) and from the call center database (cdpd). The digits are conveyed in the call ISDN message to Communication Manager and then made available to **`collect digits`** vector steps. ced and cdpd are both used to determine routing for the call.

```
1. wait-time 2 secs hearing silence
2. collect ced digits
```

```
 3. goto step 7 if digits = 1
 4. goto step 11 if digits = 2
 5. route-to number 0 with cov n if unconditionally
 6. stop
 7. collect cdpd digits
 8. route-to digits with coverage n
 9. route-to number 0 with cov n if unconditionally
10. stop
11. queue-to split 6 pri m
12. wait-time 10 secs hearing ringback
13. announcement 2564
14. wait-time 20 secs hearing music
15. goto step 13 if unconditionally
16. route-to number 0 with cov n if unconditionally
```

In this vector, step 1 provides a wait-time step if calls are transferred to this vector. Step 2 collects the ced. Steps 3 and 4 branch the call to a different vector step depending on the ced digit. If no ced were received, or if the digit received was not 1 or 2, step 5 routes the call to the attendant. If the ced digit collected was 1, the call routes to a second collect step where cdpd are collected. The vector then routes the call to the cdpd. If the ced digit collected was 2, the call queues to split 6.

CINFO interactions

Adjunct Switch Application Interface (ASAI)

Both CED and CDPD can be passed to an ASAI adjunct as collected digits with the **`adjunct routing link`** command and other event reports. ASAI passes a maximum of 16 digits.

If a TouchTone Receiver (TTR) is connected to a call as a result of ASAI-requested digit collection and the call encounters a collect ced or cdpd step, the TTR is disconnected from the call. In addition, any ASAI-requested digits stored in the collected digit buffer are discarded and no entered digits event report is sent.

ASAI does not distinguish between CINFO digits and user-entered digits that are collected as a result of a **`collect digits`** step. CINFO digits are provided to an ASAI adjunct in the same manner as any other collected digits from a vector.

The call offered to VDN domain event report contains the digits from the most recent collect ced or collect cdpd vector step.

Best Service Routing (BSR)

BSR digits are included with the call if a multisite BSR application routes the call to another Communication Manager.

Call Management System (CMS)

You do not have to enable the **Vectoring (CINFO)** field on the System-Parameters Customer-Options screen for ced or cdpd to be passed to CMS. Any version of the CMS accepts ced or cdpd.

Conference

When a conference is established, CINFO digits are merged into the call record of the conference. However, there is no indication of the party to which the digits were originally associated.
For security reasons, the CINFO digits are erased when the first ISDN call drops out of the conference.

Look-Ahead Interflow (LAI)

CINFO digits are included with the call if LAI routes the call to another Communication Manager.

Transfer

If a call is transferred from Communication Manager, CINFO digits are lost. If a call is transferred to an internal extension, CINFO digits are retained.

Important:

If a call is transferred to a VDN, the CINFO digits must not be collected until the transferring party completes the transfer. Therefore, when transfers are likely, include an appropriate **`wait-time`** step before the **`collect`** step.

Attendant Vectoring

With Attendant Vectoring, you can use a set of commands to write call vectors for calls to be routed in non-call center environments. When you set the **Attendant Vectoring** field to `y`, all attendant-seeking or dial 0 calls are processed using call vectors, and not by the normal attendant console call routing.

Attendant Vectoring is available in non distributed attendant environments and distributed attendant environments for IAS and QSIG CAS.

The main reason to use Attendant Vectoring is to allow flexible routing of attendant-seeking calls. If users are instructed to dial an attendant VDN, the call can be answered by an attendant, but the call can also be covered to the voice mailbox of a night station.

If you use Attendant Vectoring and Night Service to route calls to a voice mail system, you can also use Automatic Message Waiting (AMW) to notify after-hours personnel about messages in the night service station mailbox by assigning an AMW lamp on the backup telephones. When personnel see new messages, the personnel can check the messages and respond accordingly.

Attendant queue

If attendant vectoring results in putting a call in the attendant queue, it is placed in queue with the priority as administered on the console parameter screen. There are no changes made to the attendant priority queue for attendant vectoring. Even when partitioning is turned on and multiple attendant groups exist, all queues have the same priority assignments. Priority queue administration also applies for calls to an individual attendant, by way of the assigned extension.

Attendant VDNs

The fact that VDN extensions can be dialed directly or calls can be transferred to VDN extensions is unchanged for attendant VDNs.

Currently, VDN extensions can be assigned to the following night destinations:

Hunt group

An attendant vectoring VDN can be assigned as the night destination of a hunt group. Calls to the hunt group when it is in night service are redirected to the VDN and attendant vectoring applies. Hunt group night service does not apply if the hunt group is vector controlled. When **vector** on the Hunt Group screen is `y`, the **night service destination** field is removed from the screen. In order for a hunt group to be available in vectoring for the **`queue-to hunt-group`** command, the hunt group must be vector controlled. The hunt group in the **`route-to`** command can be in night service and the call can terminate to the indicated night service destination. If the hunt group is accessed using the **`queue-to hunt-group`** command no night service applies.

LDN and trunk

One or all trunk groups can be placed into night service and an attendant vectoring VDN can be assigned as the night service destination of the LDN or trunk group. If a night destination is assigned for LDN calls, it overrides the trunk night destination of the trunk group. Either of these destinations can be an attendant vectoring VDN. However, if Tenant Partitioning is administered and the trunk group night service destination is the attendant group, the call is redirected to the VDN associated with the tenant number of the trunk group. If, instead, the night service destination is explicitly assigned to a particular attendant vectoring VDN, it cannot be the resulted VDN had the night destination been the attendant group.

Trunk group incoming

The incoming destination can be an attendant vectoring VDN except for RLT trunk groups. As in trunk group night service, an assigned incoming destination to an attendant vector can result in the call being sent to a different VDN than if the destination had been assigned to the attendant group.

Last coverage point in a coverage path

An attendant VDN can be assigned as a coverage point. If an attendant VDN is assigned as a coverage point, the VDN be the last point in the coverage path.

Abbreviated dialing lists

Attendant VDNs can be assigned to abbreviated dialing lists.

Emergency access redirection

An attendant VDN can be assigned to emergency access redirection. When the attendant's emergency queue overflows or when the attendant group is in night service, all emergency calls are redirected to this VDN. Careful thought must be given to routing the calls off-switch.

QSIG CAS number for attendant group calls

An attendant VDN can be assigned the QSIG CAS number which determines where attendant group calls at a QSIG branch are processed. This allows local vectoring at a branch prior to routing the calls to the main or elsewhere.

Auxiliary data for the following button assignments

In keeping with existing procedures, attendant VDNs is not denied as auxiliary button data for:

- Facility busy indication. Visual indication of busy or idle status for the associated extension.
- Manual message waiting indication. Lights a message waiting lamp on the station associated with the button.

- Manual signaling. Rings the station associated with the button.
- Remote message waiting indicator. Message waiting status lamp automatically lights when a LWC message is stored in the system for the associated extension.

Attendant Vectoring and attendant VDNs

When Attendant Vectoring is administered and if an attendant VDN is assigned, attendant group calls are intercepted and sent through vector processing. The attendant VDN can be assigned on the Console Parameters screen if Tenant Partitioning is turned off or on the Tenant screen if partitioning is turned on. If an attendant VDN is assigned, the call is redirected to the VDN for vector processing. If a VDN is not assigned, the call is directed to the attendant group. Attendant group calls can only be redirected to attendant VDNs.

Intercept attendant group calls

When calls are placed to the attendant group or become attendant group calls for the reasons listed below, a check is made for an assigned attendant VDN. If an attendant VDN is assigned and either the system is not in night service or the system is in night service and a night console is available, the call is redirected to the VDN for subsequent vector processing. Otherwise, the call is treated with typical attendant group procedures.

The following occurrences can cause a call to become an attendant group call:

- Listed Directory Number (LDN)
- Attendant group in coverage path
- Attendant control of trunk group access
- Calls forwarded to attendant group
- Controlled Restriction
- Dialed attendant access code
- DID/Tie/ISDN intercept treatment
- DID time-out due to Unanswered DID Call Timer expiration
- DID busy treatment
- Security Violation Notification (SVN)
- Multi frequency signaling with attendant group as terminating destination
- CDR buffer full with attendant group as Call Record Handling Option
- Trunk incoming destination is attendant group
- Trunk group night service destination is attendant group
- Hunt group night service destination is attendant group
- Automatic Circuit Assurance (ACA) referral
- VDN routes to the attendant access code.

Vector override always applies to attendant VDNs.

Allow override

VDN override always applies to attendant VDNs.

To provide the most flexibility possible, there are no restrictions placed on the vector that is assigned to a VDN. A non attendant vector can be assigned to an attendant VDN and an attendant vector can be assigned to a non attendant VDN. Obviously, doing so is not recommended. Assigning an attendant vector to a non attendant VDN severely restricts processing for basic call vectoring since only limited vectoring commands are available in attendant vectors. Assigning a non attendant vector to an attendant VDN also severely restricts attendant vectoring since the attendant-specific commands are not available in basic call vectoring. In addition, it removes basic call vectoring information from attendant VDNs. Also, there are no restrictions in vector chaining between attendant and non attendant vectors (for example, using the **`goto vector`** or **`route-to number`** commands).

Interflow between vectors

When calls interflow from one type of vector processing to another, the calls are removed from the queue and treated as new calls to vectoring, not continuations of vectoring.

Tenant Partitioning assignments apply to attendant and non attendant VDNs. Therefore, care must be taken that a VDN assignment on the Partitioning screen has a compatible TN number assigned to the VDN. For example, tenant partition 1 can be assigned a VDN which belongs to tenant partition 2 so long as partition 1’s permissions allow access to partition 2. However, music source selection is based on the tenant partition where the VDN is assigned rather than the partition to which the VDN belongs.

Music source

When music is to be provided for attendant vectored calls, the source that is assigned to the tenant partition of the attendant seeking call is used rather than the source that is assigned to the partition of the VDN.

Console Parameters screen example

When Attendant Vectoring is enabled, a field on the Console Parameters screen identifies the assigned Attendant Vectoring VDN. The following examples show the Console Parameters screens.

Console Parameters screen (Page 1)

```
change console-parameters                                       Page   1 of   4
                              CONSOLE PARAMETERS
         Attendant Group Name: OPERATOR
                          COS: 1                                COR: 1
       Calls in Queue Warning: 1                  Attendant Lockout? y
        Ext Alert Port (TAAS): 01A1216
                          CAS: none
                                              Night Service Act. Ext.: 195
                 IAS (Branch)? n              IAS Tie Trunk Group No.:
         IAS Att. access Code:                  Alternate FRL Station:
              Backup Alerting? y        DID-LDN Only to LDN Night Ext? n

Attendant Vectoring VDN: 2000
```

Console Parameters screen (Page 2)

```
change console-parameters                                      Page   2 of   4
                              CONSOLE PARAMETERS

TIMING
  Time Reminder on Hold (sec): 30          Return Call Timeout (sec): 30
  Time in Queue Warning (sec): 15

  INCOMING CALL REMINDERS
      No Answer Timeout (sec): 10                     Alerting (sec): 10
                               Secondary Alert on Held Reminder Calls? y

ABBREVIATED DIALING
     List1:                 List2:                  List3: system

                           COMMON SHARED EXTENSIONS
            Starting Extension: 670               Count: 3
```

Console Parameters screen (Page 3)

```
change console-parameters                                      Page   3 of   4
                              CONSOLE PARAMETERS

QUEUE PRIORITIES

               Emergency Access: 1
                Assistance Call: 2
                        CO Call: 2
               DID to Attendant: 2
                       Tie Call: 2
            Redirected DID Call: 2
                Redirected Call: 2
                    Return Call: 2
                    Serial Call: 2
    Individual Attendant Access: 2
                Interpositional: 2
       VIP Wakeup Reminder Call: 2
             Miscellaneous Call: 2

       Call-Type Ordering Within Priority Levels? n
```

Console Parameters screen (Page 4)

```
change console-parameters                                      Page   4 of   4
                              CONSOLE PARAMETERS

ASSIGNED MEMBERS ( Installed attendant consoles )
              Type       Grp  TN           Type       Grp  TN
           1: principal   1   1          9:
           2:                           10:
           3:                           11:
           4:                           12:
           5:                           13:
           6:                           14:
           7:                           15:
           8:                           16:
```

Hunt group queue for Attendant Vectoring

If attendant vectoring results in putting a call in the hunt group queue, it is placed in the queue with the indicated priority. To use this command, the hunt group must be vector controlled.

Night service

There is no additional Night Service functionality provided for Attendant Vectoring. Night service routing can be provided using the existing night station service in conjunction with Attendant Vectoring. All existing night service rules remain in place, for example, night console service supersedes night station service, which supersedes TAAS. Attendant group calls are not redirected to Attendant Vectoring when the system is in night service unless a night console is available. Otherwise, the calls continue to be redirected to the applicable night service processing. To achieve Attendant Vectoring for calls when the system is in night service without a night console, the night station service extensions must be Attendant Vectoring VDN extensions.

Redirecting calls to attendant VDNs

The following types of calls cannot be redirected to the attendant VDN.

Type of calls	Description
Emergency Access	These calls are still sent directly to the attendant group. However, an attendant vectoring VDN can be assigned as the emergency access redirection extension.
Attendant Return	These calls are still sent to the original attendant if the original attendant is available or will be placed into the attendant group queue if no attendants are available.
Serial	As with Attendant Return calls, Serial calls are still returned to the original attendant if the original attendant is available and are placed into the attendant queue if no attendants are available.
VIP Wakeup	These reminder calls are still sent directly to the attendant group.
Call Park Timeout	These calls result in a conference (caller, principal, and attendant) and Call Vectoring does not allow conferenced calls to be vectored.
Call Transfer Timeout	These calls are controlled by the attendant return call timer and are processed as though the calls are attendant extended calls, in other words, actual Attendant Return calls.

Restrictions for attendant and non attendant vectoring

No restrictions apply to attendant and non attendant vectoring. For example, an attendant VDN can point to a non attendant vector and vice versa. The same is true for vector commands. For example, an attendant VDN that points to an attendant vector can have a vector step that routes to another non attendant VDN. In this case, the call is removed from the queue and treated as though it just entered vector processing rather than as a continuation from one VDN to another. The reverse is also true if a non attendant VDN is routed to an attendant VDN.

TN assignments

Just as Tenant Number (TN) assignment determines the attendant group to which calls are terminated, the TN assignment also determines the VDN to which calls are redirected. If a VDN is administered, attendant group calls are redirected to the VDN rather than the attendant group. If a VDN is not assigned, calls terminate to the associated attendant group.

The selected TN for calls that are covered to an attendant group is the called user's TN, not the calling user's TN. When Tenant Partitioning is not administered, the system can have only one partition and attendant group. All attendant group calls are directed to attendant group 1. The screen to administer TN associations is not accessible, so system-wide console assignments apply. To follow the existing principals of this administration, the attendant vectoring VDN assignment appears on the Console Parameters screen when partitioning is turned off. When it is turned on, the field is removed from the console screen and the contents are automatically copied to TN 1.

Note:

Covered calls can be rerouted to an attendant group for a different tenant partition B when the call does not queue to the attendant group for tenant number A or for the out of hours case, and so on. The vector assigned to the attendant vectoring VDN for tenant A needs to include a failure branch to a "route-to ldn_number with cov y if unconditionally" step to route to the LDN extension of tenant B. The "with coverage" parameter of the route-to step must be set to "cov y" and the original coverage path that covers to the tenant A attendant vectoring VDN must have the "Cvg Enabled for VDN Route-to party?" field set to "y".

Vector screen example

The following example shows the Call Vector screen with the Attendant Vectoring field enabled.

Call Vector screen

```
change vector xxx                                        page 1 of 3
              CALL VECTOR

    Number: xxx                  Name: ___________________________
Multimedia? n
Attendant Vectoring? y
     Meet-me Conf? y           Lock? y
     Basic? n    EAS? n   G3V4 Enhanced? n   ANI/II-Digits? n   ASAI Routing? n
 Prompting? n    LAI? n  G3V4 Adv Route? n   CINFO? n   BSR? n   Holidays? n

01 ____________
02 ____________
03 ____________
04 ____________
05 ____________
06 ____________
07 ____________
08 ____________
09 ____________
10 ____________
11 ____________
```

The **Attendant Vectoring** field appears only when Attendant Vectoring is enabled on the Customer Options screen. If either Basic Vectoring or Prompting are set to y, the Attendant Vectoring field defaults to n. If Basic Vectoring, Prompting, and Enhanced Conference are not enabled on the Customer Options screen, the Attendant Vectoring field defaults to y, and it cannot be changed to n. When the Attendant Vectoring field on the Call Vector screen is set to y, that vector is used as an attendant vector.

To associate VDNs and vectors for attendant vectoring, a field on the VDN and the call vectoring screens indicates attendant vectoring. When attendant vectoring is indicated for VDNs and vectors, all call center-associated fields (such as Skills and BSR) are not displayed.

Best Service Routing

Best Service Routing (BSR) is a Call Center Elite feature that provides singlesite and multisite load balancing and maximizes the agent staffing resources. BSR provides the best service to callers by comparing skills and routing calls to the best skill. To respond to changing conditions and operate efficiently, BSR monitors the status of the specified resources and adjusts call processing.

You can configure BSR for singlesite or multisite operations. Use singlesite BSR to compare local skills. Use multisite BSR to compare local and remote skills and to route calls to the skill that can provide the best service.

To use Best Service Routing (BSR) on a single Communication Manager, use special commands and command elements that are part of Call Vectoring. As a result, BSR for a single location can be easily added to existing vectors without modifying other parts of Communication Manager.

Multisite applications work similarly, but require additional administration. Since steps in a multisite BSR vector contact more than one remote locations, you must define the locations, administer Communication Manager to contact each location, and set up VDNs and vectors to handle communication between the sending Communication Manager and each remote Communication Manager.

Use three VDN or vector pairs in every multisite BSR application. The primary VDN or vector pair, on the sending Communication Manager, contacts the specified remote Communication Manager, collects information, compares the information, and delivers or queues the call to the resource that is most likely to provide the best service. Two VDN or vector pairs are required on each remote Communication Manager. A status poll VDN or vector pair provides information on the best resource in response to inquiries from BSR applications on other Communication Manager. Finally, you require an interflow VDN or vector pair to receive and process the calls that interflow from BSR applications on other Communication Manager.

Benefits of BSR

Singlesite and multisite BSR compare resources to find the resource that can provide the best service to a caller. With multisite BSR, you can integrate a network of call centers for better load balancing and optimal agent utilization.

Note:

If a call center network is overloaded and a significant number of calls are being blocked or abandoned, the use of BSR does not result in shorter wait times. Rather than reducing wait times, any productivity gains allows more calls to gain access to the network.

Benefit	Feature
Increased revenue	• Better agent utilization, allowing more calls to be handled with a given staff level. • Lower abandonment rates by balancing the load between resources, BSR reduces extremes in wait times across local resources or across an entire network. • In call centers with EAS, the ability to deliver calls to the best qualified or highest revenue generating agents.
Lower costs	• Better agent utilization. • Shorter trunk holding times. • Reductions of ineffective interflows. • Operation over ISDN-BRI trunks and public networks.
Improved customer satisfaction	• Interflowing calls from centers with a surplus of calls to centers with a surplus of agents. You can achieve uniform service levels across your network. This means that all callers for a given application experience approximately equivalent waiting times. • Shorter wait times. • In call centers with EAS, the ability to deliver calls to the best qualified or highest revenue generating agents. • Robust information forwarding capabilities. Multisite BSR can forward original service requirements and any caller-entered digits with each call and can use both QSIG and non-QSIG information transport methods over private or public networks.
Increased performance and more efficient trunk usage	• Less messaging and processing required per call than in traditional LAI scenarios. • Eliminates phantom calls to remote agents. • Intelligent interflows that only route calls to centers with available agents.
BSR easy configuration	• Simple vector commands. You do not have to learn complex programming languages or design comparison steps. All that you have to do is list the local and remote resources to be considered for calls and instruct Communication Manager to queue or deliver the call to the best resource on the list.
Improved agent productivity	• Increased efficiency. Improve service without adding or reducing staff while maintaining the current level of service. Network-wide load balancing means that agents at one location are less likely to sit idle while calls wait in queue at another location. • No call delivery delays. In contrast to approaches that queue calls at all remote centers simultaneously, with BSR there is no delay in delivering a call when an agent becomes available.

Table continues…

Benefit	Feature
Increased operating flexibility, easier staffing and scheduling	• Larger pool of agents available to take calls in a split or skill. Through its network-wide call distribution and information forwarding, BSR effectively converts distributed locations into a virtual call center. Staffing problems therefore, do not have to be solved on a center-by-center basis. BSR can automatically react to staff shortages at one center by routing more calls to other locations. • Automatic management of sudden and unexpected increases in call volume. Large increases in call volume for a single split or skill can be distributed across other splits or skills. Spikes in call volume at a single call center can be distributed across all call centers, provided trunk capacity is available between servers.
Improved service levels	• Lower Average Speed Answer (ASA).

BSR terminology

Term	Description
adjusted EWT	Expected Wait Time (EWT) plus a user adjustment set by a `consider` command.
agent selection method	The method that Communication Manager uses to select an agent in a hunt group when more than one agent is available to receive the next call. Possible methods are: • UCD-MIA • UCD-LOA • EAD-MIA • EAD-LOA The agent selection method is a property of hunt groups and is set in the **Group-Type** field on the Hunt Group screen. **Note:** To use an EAD available agent strategy, you must enable **Expert Agent Selection (EAS)**.
Alternate Selection on BSR Ties	BSR compares splits or skills using a series of `consider` steps and selects the one that provides the best service to a call. When that comparison results in a tie, the Alternate Selection on BSR Ties determines how BSR chooses which agent, skill, or location to select.
application	A general term for a system in any call center that handles calls of a particular type. In relation to BSR, any specific implementation of multisite BSR.
application plan	Used only in multisite applications, the application plan identifies the remote switches that can be compared in consider series. The plan also specifies the information that is used to contact each switch and to interflow calls to it.

Table continues…

Term	Description
best	Includes the following conditions • No agents available - When no agents are available in any of the specified splits or skills, the best resource is the one with the lowest adjusted EWT. • Agent available in one resource - When an agent is available in one and only one of the splits or skills that are specified in a consider series, that agent is the best and the call is delivered to that agent. If the **BSR Available Agent Strategy** field is set to `1st-found`, BSR ignores all subsequent steps in the consider series. If any other available agent strategy is used, all remaining resources are still checked before the call is delivered. • Agents available in more than two resources - When agents are available in more than two splits or skills, the best agent is the one that best meets the criteria that are specified in the **BSR Available Agent Strategy** field. For example, if the available agent strategy is UCD-MIA, the best agent out of those available is the agent with the longest idle time.
Best Service Routing (BSR)	A feature that is based on Call Vectoring and routes ACD calls to the resource that is best able to service each call. BSR can be used on a single switch, or to integrate resources across a network of switches.
BSR available agent strategy	A field that appears on the VDN screen when either version of BSR is enabled. The entry in this field is a property of the VDN and its assigned vector. Possible entries are: • `1st-found` • `UCD-MIA` • `UCD-LOA` • `EAD-MIA` • `EAD-LOA` When the VDN is the active VDN for a call, as determined by VDN Override, this field determines how BSR commands in the vector identify the best split or skill when several have available agents.
consider series	**`Consider`** commands are written in a sets. This set of **`consider`** commands is called a consider series. A consider series in a status poll vector can have just one **`consider`** step.
consider sequence	A consider sequence is a consider series plus a **`queue-to best`**, **`check-best`**, or **`reply-best`** step.
Expected Wait Time (EWT)	Expected Wait Time is an estimate of how long a call in the queue has to wait before being connected to an agent.
Intelligent polling	An automatic feature of BSR that significantly reduces the number of status polls that are executed. When a remote location cannot be the best resource at a given moment in time, the intelligent polling feature temporarily suppresses polls to that location.
interflow	The process of routing an incoming call to an external switch without answering the call at the origin switch.

Table continues…

Term	Description
poll suppression	A component of BSR intelligent polling that eliminates wasteful polling of remote locations which have returned poor adjusted EWTs.
resources	An agent, split, skill, or location
status poll	A call that is placed by a **`consider location`** vector command to obtain status data from a remote location in a multisite BSR application.

BSR requirements

For singlesite BSR applications, Communication Manager must meet the requirements indicated in the following pages. The requirements for ISDN trunks and Look-Ahead Interflow (LAI) do not apply to singlesite BSR applications.

To use multisite BSR applications, all servers involved and the network connecting the servers must meet all the requirements that are described in this section.

Caution:

To ensure that the network meets the requirements for BSR support, contact your Account Executive about the BSR network certification.

Server requirements

Communication Manager must meet the requirements shown in the following table to support BSR.

Screen name	Page number	Field name	Field setting
System- Parameters Customer- Options	4	**ISDN-BRI Trunks** Multisite BSR operates over both BRI and PRI trunks. ISDN connectivity is necessary only to use multisite BSR, in which case set one or both the fields to `y`.	`y`
		ISDN-PRI You can set up ATM trunking, which is on page 3, and IP trunk, which is on page 4, to emulate ISDN PRI.	`y`
	6	**Vectoring (G3V4 Advanced Routing)**	`y`
		Vectoring (Best Service Routing)	`y`
		Lookahead Interflow (LAI) (To use multisite BSR)	`y`
Feature-Related System Parameters	12	**Adjunct CMS Release**	`R3V6` or later `blank`

Tip:

If you use BSR and then turn BSR off, you cannot set **Vectoring (Best Service Routing)** to `n` until you remove all BSR commands from the vectors. If you are using multisite BSR with LAI and want to turn LAI off, you cannot set **Lookahead Interflow (LAI)** to `n` until you remove all **`consider location`**, **`reply-best`**, and **`interflow-qpos`** commands from the vectors.

Network requirements

To support multisite BSR, networks must meet both the criteria for LAI call control operation over switched networks and the following criteria:

- The network must support end-to-end transport of codeset 0 user data, either as a User-to-User Information Element (UUI IE) or by a QSIG Manufacturer Specific Information (MSI IE), in the ISDN SETUP and DISCONNECT messages.
- With BSR poll calls, information is forwarded in the DISCONNECT message. In this case, the network must support forwarding of UUI in the first call clearing message, while the call is still in the call proceeding state, prior to the active state.
- Private networks can be configured for either QSIG using MSI packaged in codeset 0 Facility IEs or non-QSIG using a codeset 0 UUI IE transport. Currently, public networks cannot be configured for QSIG and user data can only be transported by the UUI IE when supported by the network. Future public network can be configured for QSIG, possibly by a Virtual Private Network (VPN).
- The switch must support the ISDN country protocol.
- The network byte limit for the user data portion of the user information contents must be large to carry the data needed for the customer application.

Note:

Some public network providers can require service activation, fees for user information transport, or both.

BSR, LAI, enhanced information forwarding, and UCID have been tested with several major carriers. To find if the capabilities work with your carrier, check with your account team for the most current information.

If testing has not been done to verify operation over the public networks that are involved with the preferred specific configuration, use private ISDN trunking between the nodes until successful testing is complete.

BSR considerations

- If more than one of the considered resources have an available agent, the resources with EWT are ignored. This means that there is an agent surplus.
- If the available agent strategy assigned to the active VDN is 1st-found, the adjust-by parameter is ignored and the first consider with an available agent is used for the queue-to best.
- If the available agent strategy is UCD-MIA, EAD-MIA, UCD-LOA, or EAD-LOA and there is more than one consider step with an available agent, the adjust-by parameter is applied as part of the algorithm to select the best of the possible choices.

Special BSR local treatment considerations

Trunk group status

Communication Manager displays the calls that queue remotely but receive local treatment as *active* trunk members if you execute the **`status trunk-group`** command on the interflowed trunk group. Even though the H.323 (IP) trunk member is *active*, no bandwidth is used because no voice packets are transmitted for local treatment.

Path replacement

Both ends of the connection must be answered for path replacement to work. When you set **BSR local treatment** to `y`, the local VDN answers, but the remote VDN where the call queues does not answer. Therefore, path replacement does not occur when local treatment VDNs queue a call remotely.

How BSR determines the best resource

BSR determines the best resource to service a call by examining one or all of the following variables:

- The Expected Wait Time (EWT) of the resource
- Any user adjustments
- The availability of agents
- The selection strategy for the active Vector Directory Number (VDN)

Note:

The BSR available agent strategy that applies to a given call is the strategy that is assigned to the active VDN for that call, as determined by VDN override.

Call surplus situation

Every BSR application compares a set of predetermined resources, splits or skills, and selects the best resource to service the arriving ACD call.

In a call surplus situation when no agents are available, the best resource is always the skill with the lowest EWT. For the purpose of calculations, the best resource in a call surplus situation, with BSR, you can adjust the EWT figure for any split or skill. The actual EWT for calls in queue is not changed. Only the figure used in the calculations performed by the BSR feature is changed. You do not have to enter adjustments, but the ability to adjust the EWT for splits or skills allows you to program preferences in vectors. Because of agent expertise, for example, or the availability or cost of tie trunks, you can choose that some resources do not service a call unless doing so significantly decreases the time in queue for the call.

You can make adjustments to agent availability using the **`consider`** step.

Agent surplus situation

In an agent surplus situation when more than one agent is available to take incoming calls, BSR delivers a new call according to the **BSR Available Agent Strategy** field specified on the VDN screen. The best resource is the split or skill that meets one of the five criteria that you have defined by the strategy as the VDN BSR Available Agent Strategy. BSR can use any of the five strategies shown in the following table to select an agent when agents are available.

Field setting	Call delivery
`1st-found`	The first available agent. BSR does not check any other resources as soon as BSR finds an available agent.
`ucd-mia`	The resource with a staffed agent who has been idle for the longest time. BSR compares all the splits or skills specified in the vector before delivering the call.
`ead-mia`	The resource with a staffed agent who has the highest (most primary) skill level that is relevant to the call and who has been idle the longest. BSR compares all the splits or skills specified in the vector before delivering the call.
`ucd-loa`	The resource with the least-occupied agent. BSR compares all the splits or skills specified in the vector before delivering the call.
`ead-loa`	The resource with an agent who has the highest (most primary) skill level that is relevant to the call and who is the least occupied. BSR compares all the splits or skills specified in the vector before delivering the call.

When agents are available in more than one of the specified resources, BSR does not check resources whether local or remote that return an Expected Wait Time value as is the case in a call queue or call surplus situation, when selecting the best place to send the call.

Note:

With the exception of the *1st-found* field setting, all other field settings must match the agent selection method used in the splits or skills checked by a BSR application.

Agent selection adjustments

An option has been provided to have the BSR adjust-by value apply in the agent surplus (agents available) situation. This adjustment provides the ability to use the **`consider`** step adjustment value to prioritize (handicap) agent resources when agents are available.

When the adjustment is used, the **`consider`** step uses the following syntax:

`consider split/location adjust-by` **`x`**

The server applies the agent adjustment in the same manner as the calls in queue/call surplus (lowest EWT) situation.

To select an adjustment, think in terms of reducing the importance of a resource/site and in relative percentage — the higher the adjustment, the less desirable it is to pick that agent/site. So, if **`x`** = 30, then the agent/site is 30% less desirable.

The available agent adjustment applies to the UCD-MIA, UCD-LOA, EAD-MIA, and EAD-LOA call distribution methods. For the most idle agent distribution methods, the adjust-by lowers the idle time value returned by the agent/site. For the least occupied agent distribution methods, the adjust-by raises the returned occupancy level of the agent/site. In either case, with EAD, the MIA or LOA is used as a tie breaker if more than one site has an agent available with the same highest skill level.

The same adjust-by value in the **consider** step applies to both agent surplus and call surplus situations.

Tips for writing BSR vectors

Before you write vectors using BSR, see the sample vectors provided and familiarize yourself with the new commands and command elements. Sample vectors are provided in singlesite BSR and multisite BSR. The new commands and command elements are explained in the Call Vectoring commands.

As you write BSR vectors, follow the guidelines:

- Arrange the **consider** steps in the order of preference.

  The **consider** step that tests the main or preferred, resource must be the first in the series. The second **consider** step must test the resource, that is, your second preference for handling the given call type. To prevent unnecessary interflows, use **consider** steps for local resources before steps for remote resources. This arrangement also provides a local best as a backup in case the interflow fails.

  Arranging **consider** steps in the order of preference for all BSR vectors. This arrangement is especially important when the active VDN for the call is using the 1st-found agent strategy since the server delivers the call to the first available agent found, arranging **consider** steps in the order of preference ensures that calls are delivered to the best available resources.
- Do not put any commands between the steps of a **consider** series that can cause a delay. **Goto** commands are ok.
- Do not put a **consider** series in vector loops.
- Confirm that calls queue successfully.

  Since EWT is infinite for a call that is queued, a step that checks EWT after a queue attempt is a good confirmation method. After a **queue-to best** step, for example a command such as **goto step X if expected-wait for call < 9999** must be included.
- Do not use the wait-improved conditional in a vector before you have queued the call once.

  The wait-improved conditional compares the EWT of the call in the current queue to the best resource found by a **consider** series. If a call is not queued and a vector step such as **check best** if wait-improved > 30 is executed, the server interprets the current EWT of the call as infinite and the **check best** step always routes the call to the best resource. In other words, in this situation the **check best** step functions like an unconditional **goto** or **route-to** command.

Alternate Selection on BSR Ties

With Best Service Routing (BSR), Communication Manager uses a series of **consider** steps to compare splits or skills and to select the split or skill that can provide the best service to a call. When that comparison results in a tie, that is, the results are equal in value, Communication Manager uses Alternate Selection on BSR Ties to select an agent, skill, or location.

Communication Manager uses Alternate Selection on BSR Ties to select between the following:

- Skills or locations with the same Expected Wait Time (EWT).
- Available agents that are weighted with the same criteria, that is, the most idle agent or the least occupied agent, in a **`consider`** series designed to search for the best skill or best location.

Administer **BSR Tie Strategy** for the system or for each Vector Directory Number (VDN). Communication Manager uses each **`consider skill/location`** step to compare the current best choice with the previous **`consider`** step to the value obtained from the current **`consider`** step.

If **BSR Tie Strategy** on the active VDN for the call is **1st-found**, Communication Manager stops processing **`consider`** steps when Communication Manager finds an available agent at a **`consider`** step. Alternate Selection on BSR Ties is inapplicable in such cases.

Singlesite BSR

Singlesite BSR is a simple, logical extension of Call Vectoring. Like any other vector, vectors with BSR commands are assigned to more than one VDN. Using new vector commands and command elements, you tell the Communication Manager to compare specific splits or skills for each call that is processed in that vector. Throughout the comparison, Communication Manager can remember which resource is the best based on how you define best. BSR vectors can deliver a call to the first available agent found, or check all the specified resources and deliver the call to the best split or skill. If no agents are available in any split or skill, Communication Manager queues the call to the split or skill with the shortest adjusted EWT.

Planning for singlesite BSR

About this task

To work efficiently, record goals, VDN extensions, vector numbers, and other information on paper before you begin your administration session. To do this, complete the following:

Procedure

1. Select the group of callers for which singlesite BSR is to be used and identify the VDNs and vectors to support the group.
2. Define your business goals.

   For example, your business goals in using BSR can be to achieve a faster Average Speed of Answer (ASA) for a certain call type, or perhaps better service by routing calls to the most qualified agents.

   Different VDNs or vectors have different goals.
3. Decide which agent selection strategy to assign to each VDN in order to achieve the goals relevant to the VDN.
4. Decide if the VDNs must follow VDN Override Rules.

Singlesite BSR command set

The following table shows the screens, the vectors, the vector commands and command elements used in singlesite BSR. The following table shows the vector commands and command elements used in singlesite BSR applications.

Commands and command elements		Purpose
Screens	Vector Directory Number	To link a VDN to a BSR vector. To set the agent selection strategy that is used for all calls to that VDN.
	Call Vector	To confirm that BSR is administered. To write vectors that use BSR commands.
Commands	**`consider split/ skill`**	To retrieve the EWT or agent data required to identify the best local resource. One **`consider`** step must be written for each split or skill that is to be checked. Since the **`consider`** command is designed to compare more than two resources, **`consider`** commands are written in a series of more than two commands with the sequence terminating in a **`queue-to best`** vector step. This set of **`consider`** commands and a **`queue-to best`** step is called a consider sequence.
	`queue-to`	With the *best* keyword to queue calls to the best resource identified by the consider sequence.
	`check`	With the *best* keyword to queue calls to the best resource identified by the consider sequence if the resource meets certain conditions.
Key word	**`best`**	Use the *best* keyword in **`queue-to`**, **`check`**, and **`goto`** commands that refer to the resource identified as *best* by a series of **`consider`** steps
Conditional	**`wait-improved`**	To prevent calls from being queued to an additional split or skill when the reduction in EWT is not enough to be useful. Wait improved means that the EWT of a call must be improved by a specific amount, specified in seconds, over the current EWT or Communication Manager does not queue the call to the additional split or skill.

Table continues…

Commands and command elements		Purpose
User adjustment	**`adjust-by`**	To specify your preferences for the splits or skills to handle the calls for a particular application, reflecting factors such as agent expertise or reducing calls to a backup split or skill. When a vector checks a local resource you can make the selection of that split or skill less desirable. The higher the setting, the lesser the chance that the resource is selected over another with a lower setting. For example, set to 30 makes that choice 30 percent less desirable. With EWT returned, the setting increases the returned EWT for comparison with other returned EWTs. As a result, the split or skill is less likely to service the call unless its EWT is significantly less than that of any other available split or skill. Optionally, the adjust-by parameter applies in the available agent case. If you are using the UCD-MIA or EAD-MIA available agent strategy, the setting decreases the returned agent idle time, making the agent appear less idle. If you are using the UCD-LOA or EAD-LOA available agent strategy, the setting increases the returned agent occupancy, making the agent appear busier. In either case with EAD, the MIA or the LOA is used as a tie breaker if more than one site has an agent available with the same highest skill level.

Example of basic singlesite BSR

The central element of singlesite and multisite BSR is a VDN or a vector pair. The vector contains the commands that process the call, but the active VDN for the call contains information that is used by the vector steps.

For singlesite BSR, the active VDN for a call sets the available agent strategy that is used by the vector.

Singlesite BSR example VDN screen

```
change vdn xxxxx                                                page 1 of 3
        VECTOR DIRECTORY NUMBER

                            Extension: 5000
                                 Name: Singlesite BSR
                        Vector Number: 234
                  Attendant Vectoring? n
                   Meet-me Conference? n
                   Allow VDN Override? n
                                  COR: 59
                                   TN: 1
                             Measured: internal
       Acceptable Service Level (sec): 20
              Service Objective (sec):
        VDN of Origin Annc. Extension:
                            1st Skill:
                            2nd Skill:
                            3rd Skill:
```

```
change vdn xxxxx                                               page 2 of 3
        VECTOR DIRECTORY NUMBER
```

```
                           Audix Name:

                   Return Destination:
               VDN Timed ACW Interval:
                      BSR Application:31
         BSR Available Agent Strategy: 1st-found
```

In the example Vector Directory Number screen, the **BSR Available Agent Strategy** field is set to `1st-found`. If vector 234 uses BSR commands, when a **`consider`** step locates a resource with an available agent, all subsequent consider steps are skipped and the call is delivered to that resource. Resources that are specified in any subsequent **`consider`** commands are not checked. If no split has an available agent, the call is queued to the split with the lowest adjusted EWT.

If the **Allow VDN Override** is set to `n` and a second VDN and vector are used to process this call, the 1st-found strategy specified in VDN 5000 is still used.

In the preceding example, VDN 5000 is associated with vector 234, which is shown below. In this example, vector 234 compares two splits. No adjustment is assigned to either resource, indicating that both splits are equally suited to service calls since neither is preferred to the other. In reality, such a vector probably has additional steps after step 4, such as **`announcement`** or **`wait-time`** commands. The steps are omitted in this example for purposes of clarity.

Singlesite BSR example vector

```
1. wait time 0 secs hearing ringback
2. consider split 1 pri l adjust-by 0
3. consider split 2 pri l adjust-by 0
4. queue-to best
```

Note that the **`consider`** commands follow each other in unbroken sequence and that the **`queue-to best`** command immediately follows the last **`consider`** command. This structure is called a *consider series*. Write such series in uninterrupted order. You can use a few commands, such as the **`goto`** command, which cause little if any delay in the execution of the **`consider`** steps. In general, however, do not put other commands between **`consider`** steps, or between a **`consider`** step and a **`queue-to best`** step. Even if BSR still works in that situation, the performance can be impaired.

Consider commands collect and compare information. When a call is processed in the vector, the first consider step collects and temporarily saves the following information about split 1:

- The fact that split 1 is a local split.
- The queue priority that is specified in the consider step.
- The user adjustment that is specified in the consider step.
- The split's
  - Split number
  - Expected Wait Time (EWT)

If EWT=0, which indicates that more than one agent is available, the step also collects all the agent information required by the BSR available agent strategy. This includes:

- Agent Idle Time (AIT)
- Agent Occupancy (AOC)
- The skill level of the agent in the split or skill who receive the next call

In the example, the splits do not have an available agent when the consider series executes. If a split has an available agent, the call is delivered to that split by the **`queue-to`** best step. Since there are no available agents in either split, the complete set of saved data now defines the best resource. The second consider step collects and compares the same data to the current best data. For this example, if the EWT for split 1 is 40 seconds and the EWT for split 2 is 20 seconds, when the second consider step executes, the data replaces the best data from step 1 because the adjusted EWT is lower. The best data is essentially a placeholder. When a **`queue-to`** best step executes, the step reads the data saved as the best at the moment and queues the call to the split. In this case, the best data was collected from split 2, so the call is queued to split 2 at the specified priority.

What if there are available agents in both splits?

As *BSR Available Agent Strategy* in this example is 1st-found, the consider series skips the consider steps after step 2 and the queue-to best step delivers the call to split 1, which is the first split or skill with an available agent that is found by the vector.

In any BSR vector, the order of the consider steps must reflect your preferences for the resources. In the consider series, put the step that considers the most preferred split or skill first, followed by the other splits or skills.

What if there are several available agents in split 1? Which agent receives the call?

When more than one agent is available in a split, the BSR **`consider`** command collects agent data only for the agent who will receive the next call to that split. This agent is identified according to the agent selection method that is specified in the **Group-Type** field on the Hunt Group screen.

Note:

For greatest efficiency, the agent selection method used in the splits or skills checked by a BSR vector must match the BSR Available Agent Strategy that is assigned to the active VDN.

User adjustments in singlesite BSR

In singlesite and multisite BSR, with the *adjust-by* portion of the **`consider`** command, you can implement your preferences into the vectors.

You can assign a value of 0 to 100 in user adjustments. The server supplies the units of the value depending on the conditions whenever the **`consider`** step executes. For example, in the command **`consider split 1 pri h adjust-by 20`**, the server interprets *adjust-by 20* to mean *add 20 percent to the EWT, but add 20 seconds at the minimum*.

Note:

If you define the user adjustment as a number of seconds, BSR is not efficient when EWT was high. If you define the user adjustment as a percentage, BSR is not efficient when EWT was low. Such efficiencies, while always important, become critical in multisite BSR applications where issues of trunk cost and capacity are involved.

For EWT of 1 to 100 seconds, an adjustment of 20 adds 20 seconds. Over 100 seconds, the same adjustment adds 20 percent to the EWT for the split or skill that is specified in the **`consider`** step. The following table shows the results of user adjustment to a range of EWT.

EWT of resource (in seconds)	User adjustment	Adjustment applied by the server (in seconds)	Adjusted EWT used to select resource
10	20	20	30
60		20	80
120		24	144
300		60	360

Example of singlesite BSR with adjustments

The following example shows a more complex implementation of singlesite BSR. Four skills in an EAS environment are compared. The Expected Wait Time (EWT) for some skills is adjusted to reflect the administrator preferences.

Singlesite BSR example VDN screen

```
change vdn xxxxx                                                              page 1 of 3
                    VECTOR DIRECTORY NUMBER

                            Extension: 5001
                                 Name: Singlesite BSR
                        Vector Number: 11
                  Attendant Vectoring? n
                   Meet-me Conference? n
                   Allow VDN Override? n
                                  COR: 59
                                   TN: 1
                             Measured: internal
       Acceptable Service Level (sec): 20
              Service Objective (sec):
        VDN of Origin Annc. Extension: 501
                            1st Skill:
                            2nd Skill:
                            3rd Skill:
```

```
change vdn xxxxx                                                               page 2 of 3
                  VECTOR DIRECTORY NUMBER

                           Audix Name:
                   Return Destination:
               VDN Timed ACW Interval:
                      BSR Application: 19
         BSR Available Agent Strategy: EAD-MIA
```

In the example, the **BSR Available Agent Strategy** field is set to `EAD-MIA`. If vector 11 uses BSR commands, calls are not automatically delivered to the first resource with an available agent that is found. All **`consider`** steps in vector 11 are executed and one of the following things happens:

Condition	Action
No skill has an available agent	The call queues to the skill with the lowest adjusted EWT.
One skill has an available agent	The call is delivered to the skill.
More than two skills have available agents	The call is delivered to the skill with the most expertise.
More than two skills have available agents with the same skill level	The call is delivered to the agents who has been idle the longest.

Also note that **Allow VDN Override** is set to `n`. If a second VDN and vector are used to process this call, the EAD-MIA strategy that is specified in VDN 5001 is used. If **Allow VDN Override** is set to `y` and vector 11 routes some calls to another VDN, the available agent strategy of the subsequent VDN governs the operation of **`consider`** steps in its vector.

The following example vector 11, which compares four skills.

Singlesite BSR example vector

```
1. wait-time  0 secs hearing ringback
2. consider skill  1 pri l adjust-by 0
3. consider skill  2 pri l adjust-by 30
4. consider skill 11 pri l adjust-by 30
5. consider skill 12 pri l adjust-by 30
6. queue-to best
7. wait-time 10 secs hearing ringback
8. announcement 1001
9. wait-time 30 secs hearing music
10. goto step 8 unconditionally
```

For this example, the EWT of the four skills are 95, 60, 180, and 50 seconds, respectively. Note that all **`consider`** steps except the first adjust the EWT returned by the specified skill. Skill 1 is the preferred skill to handle calls to VDN 5001, so the EWT is not adjusted. Skills 2, 11, and 12 can handle this call type, but the skills are not preferred. The adjustment of 30 means that, in call surplus situations, the skills do not handle calls to VDN 5001 unless the EWT is a minimum of 30 seconds better than the EWT in skill 1.

The following table shows the adjustments to be applied to each skill given the EWT and the user adjustment specified in the **`consider`** step. The last column shows the adjusted EWT the server uses to select a skill for the call.

User adjustments				
Skill number	**User adjustment in the consider step**	**Actual EWT (in seconds)**	**Adjustment applied by the server (in seconds)**	**Adjusted EWT used in BSR calculations (in seconds)**
1	0	95	0	95
2	30	60	30	90
11	30	180	54	234
12	30	50	30	80

Since the available agent strategy is not 1st-found, all four **`consider`** steps are executed each time that the vector processes a call. In this example, there are no available agents in any skill. In fact, EWT is high in the first three skills for the server to queue the call to skill 12.

When the **`queue-to-best`** step executes, the data in the best data placeholder is the data from skill 12 and so the call is queued to that skill. From this point on, if the call is not answered during the execution of step 7, a common vector loop regularly repeats an announcement for the caller while the caller waits in the queue.

User adjustments also apply to available agent situations, with a strategy other than first found, in a manner similar to EWT.

What if there is an available agent in one skill?

Since the “BSR Available Agent Strategy” in this example is EAD-MIA, the entire consider series will always be executed to check all of the skills for available agents. If only one skill has available agents, the call is delivered to that skill and user adjustments are not applied.

What if there are available agents in two skills?

Since the BSR Available Agent Strategy for VDN 5001 (the active VDN) is EAD-MIA, the call is delivered to the skill with the most expert agent. If there are available agents in both skills with the same skill level, their user adjusted idle times are compared and the call goes to the skill with the agent who has the longest adjusted idle time.

If a split/skill has more than one available agent, remember that it is the split/skill’s agent selection method that determines which agent’s data is used in BSR selection of the best resource.

What if no agents are staffed in a skill? Will the server recognize this?

Yes. Under any of the following conditions, the EWT returned from a split/skill is infinite:

- No agents logged in
- No queue slots available
- All agents in AUX work mode

The server logs a vector event and goes to the next vector step without changing the data in the best placeholder. A resource with an infinite EWT is never selected as the best resource.

Can VDN skills be used in consider steps?

Yes. For example, consider skill 1st [2nd, 3rd] pri m adjust-by 0 will collect data on the 1st [2nd, 3rd] skill, as defined for the active VDN.

Multisite BSR

Multisite BSR extends all of the capabilities of singlesite BSR across a network of Communication Manager. Multisite BSR compares local splits/skills and remote splits/skills, and routes calls to the resource that provides the best service.

Multisite BSR has special features that work to ensure efficient use of processor power and network resources in your BSR applications.

Multisite BSR command set

The following table shows the screens, vectors, special vector commands, and command elements required to administer multisite BSR applications.

Screen name	Description
Best Service Routing Application Plan	• To define the group of remote sites to be polled by a specific application. • To assign a unique name and number to each application. • To assign routing numbers for the status poll and interflow VDNs.

Table continues…

Screen name	Description
Vector Directory Number	• To link a VDN to a BSR application by the application number. • To link the VDN to a BSR vector. • To set the agent selection strategy to be used for all calls to that VDN.
Call Vector	• To confirm that BSR is administered and to program the vector steps for BSR.
ISDN Trunk	• To administer Communication Manager to forward user information by Shared UUI or QSIG MSI.
List Best Service Routing Applications	• To display a list of all the BSR applications by name and number.
System Capacity	• To monitor the number of BSR application-location pairs assigned in the system.
VDNs and Vectors	
Primary VDN, which is the active VDN for the call at the origin, as determined by VDN Override Rules	• To define the application plan and available agent strategy used by the vector assigned to this VDN.
Primary vector	• To control call processing at the original server and compare local and remote resources.
Status poll VDN or vector	• To respond to status poll calls from another server. The status poll vector checks a set of local splits or skills and returns data on the best resource to the original server.
Interflow VDN or vector	• To accept BSR calls from another server and to queue the calls to the best of the local resources.
Commands	
consider split/ skill	• To obtain the EWT or agent data needed to identify the best local resource. One **consider** step must be written for each split or skill that is to be checked. Since the **consider** command is designed to compare more than two resources, **consider** commands are written in a series of more than two commands with the sequence terminating in a **queue-to best** vector step. This set of **consider** commands and a **queue-to best** step is called a consider sequence.
consider location	• To obtain the EWT or agent data required to identify the best resource at a remote server. One **consider** step must be written for each location that is to be checked. Routing information is obtained from the BSR Application plan for the active VDN.
reply-best	• To return data to another server in response to a status poll.
queue-to	• With the *best* keyword to queue calls to the best resource identified by the consider sequence.
check	• With the *best* keyword to queue calls to the best resource identified by the consider sequence if the resource meets certain conditions.
Key word	

Table continues…

Screen name	Description
best	• In **queue-to**, **check**, and **goto** commands that refer to the resource identified as *best* by a series of **consider** steps
Conditional	
wait-improved	• To prevent calls from being queued to an additional split or skill that can be local or remote when the reduction in EWT is not useful. Wait improved means that the EWT of a call must be improved by a specific amount, which is a figure that you specify in seconds, over the current EWT or the server does not queue the call to the additional split or skill.
User adjustment	
adjust-by	• To control long distance costs and limit trunk usage, reflecting factors such as availability of the trunks or agent expertise at remote locations. When a vector polls a local or remote resource, you can make the selection of that site less desirable. The higher the setting, the lesser the chance that the resource is selected over another with a lower setting. With EWT returned, the setting increases the returned EWT for comparison with other returned EWTs. Optionally, the *adjust-by* setting applies in the available agent case. If you are using the UCD-MIA or EAD-MIA available agent strategy, the setting decreases the returned agent idle time, making the agent appear less idle. If you are using the UCD-LOA or EAD-LOA available agent strategy, the setting increases the returned agent occupancy, making the agent appear more occupied. In either case with EAD, the MIA or the LOA is used as a tie breaker if more than one site has an agent available with the same highest skill level.

Multisite BSR applications

You can implement BSR at a single location by using the BSR commands in vectors. Using BSR across a network is more complex and requires additional administration.

A series of consider location steps in a multisite BSR vector contacts several remote locations. You must define the locations, determine how the server contacts each location, and set up VDNs and vectors to handle the communication between the origin server and the remote servers.

Note:

You can use any combination of split or skill numbers, VDN numbers, and vector numbers to support a single customer application or call type across a network. For clarity and simplicity, use the same BSR Application Plan number and the location numbers on all servers for a given application.

You must also set up the ISDN trunk groups, parameters for information forwarding (UUI Transport), and administer the numbering plans and the AAR/ARS tables.

Multisite BSR starts with the active VDN for a call, as determined by VDN Override rules. If you want any specific VDN or vector pair to interflow calls using multisite BSR, you must create a specific application for the VDN or vector. A multisite application must contain the elements shown in the following table.

BSR application elements	Purpose
The primary VDN	The primary VDN is the active VDN for a call at the origin server, as defined by the VDN Override rules. Therefore, the primary VDN in a BSR application does not have to be the VDN that originally received the incoming call. The primary VDN links its assigned vector to a BSR application plan and sets **BSR Available Agent Strategy**.
The primary vector that handles the incoming call on the origin server	The primary vector contacts the specified remote servers, collects information, compares the information, and delivers or queues the call to the resource that is likely to provide the best service.
An application plan	The application plan identifies the remote servers and specifies the information to be used to contact each server and to route calls.
Two VDN or vector pairs on each remote server: • Status poll VDN or vector • Interflow VDN or vector	Status poll VDN or vector The status poll vector compares splits at the location and replies to the origin server with information on the best of the splits. Each remote server in a given application must have a dedicated status poll VDN or vector.
	Interflow VDN or vector When a given remote server is the best available, the origin server interflows the call to the VDN or vector on the remote server. Each remote server in a given application must have a dedicated interflow VDN or vector. The steps in this vector deliver or queue the call to the best resource found by the status poll vector.

To create a multisite BSR application, first create an application plan on the origin server.

Note:

Remember that the terms local, origin, and remote are relative terms. In most networks that use multisite BSR, every server can interflow calls to other servers and receive interflowed calls from other servers. Therefore, every server in the network can have all the elements described in the table. For clarity in the following discussions, local or origin means a server that decides whether to interflow a call. Remote means a server that is polled by the origin server.

Application plans

The application plan identifies the remote servers and specifies the information that is used to contact each server and to route calls.

You can identify the plan for each application by the application number and a name. The plan specifies the remote servers to be polled by the application and identifies each with a number called the Location Number. The plan also specifies the numbers for the status poll and interflow VDNs for each remote server. Whatever you dial to reach the VDNs is what you must enter in the fields: full length numbers as well as AAR, ARS, UDP, or public network numbers work.

Create application plans on the Best Service Routing Application screen. A plan for an application with three remote servers looks like the following example.

Sample multisite BSR application plan

```
BEST SERVICE ROUTING APPLICATION PLAN
Number: 15    Name: Customer Service    Maximum Suppression Time: 60     Lock? y

Num  Location      Switch   Node          Status Poll VDN     Interflow VDN    Net Redir?
1    New Jersey    320      84015         84115                                 n
2    Denver        18       913031234015                  913031234115          n
4    New York      12345    912121234015                  912121234115          n
___ ______________      ____  ____   __________  __     ____________            n
___ ______________      ____  ____   __________  __     ____________            n
___ ______________      ____  ____   __________  __     ____________            n
___ ______________      ____  ____   __________  __     ____________            n
___ ______________      ____  ____   __________  __     ____________            n
___ ______________      ____  ____   __________  __     ____________            n
___ ______________      ____  ____   __________  __     ____________            n
___ ______________      ____  ____   __________  __     ____________            n
___ ______________      ____  ____   __________  __     ____________            n
___ ______________      ____  ____   __________  __     ____________            n
___ ______________      ____  ____   __________  __     ____________            n
```

The maximum number of application plans vary depending on your Communication Manager software release and platform. For more information, see *Avaya Aura® Communication Manager System Capacities Table*.

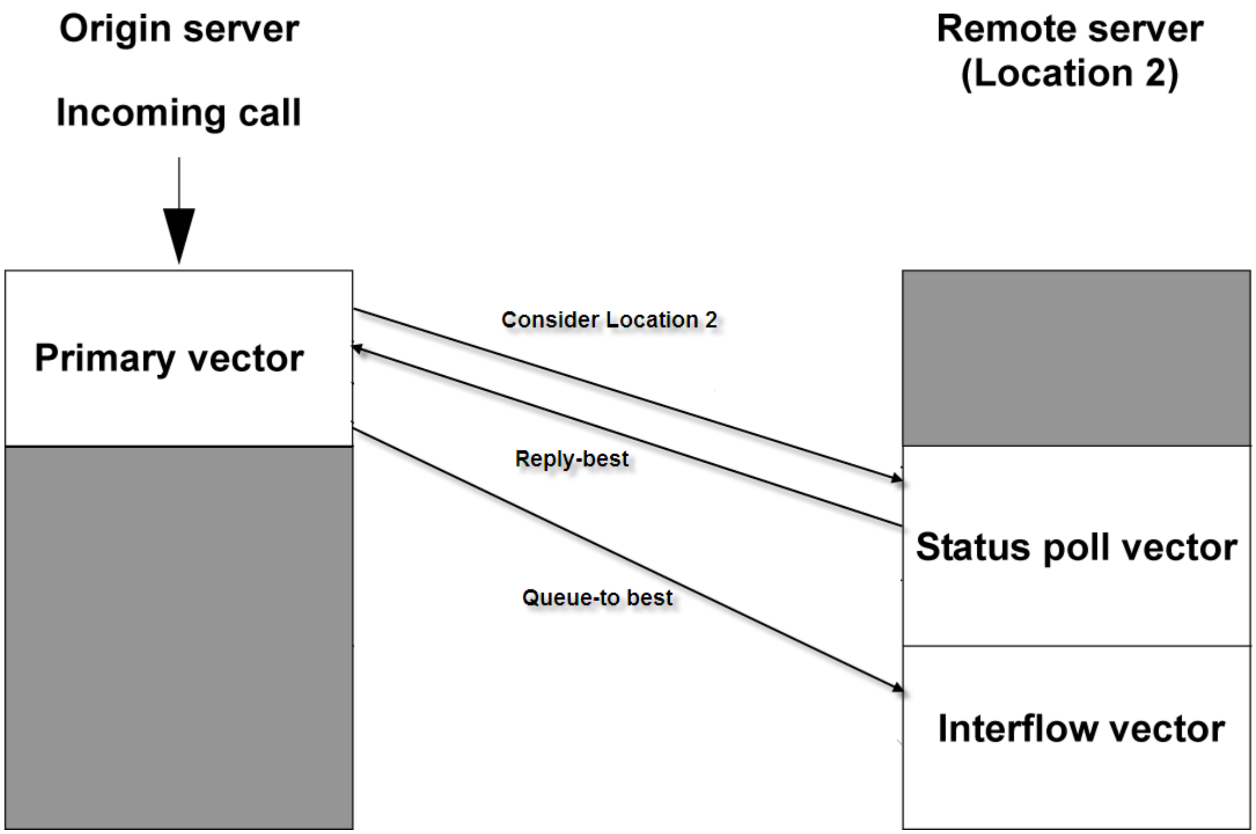
By entering the application number from this plan on a VDN screen, you can link a given VDN on your local server to this list of locations. This VDN becomes the primary VDN for the application. For example, if the primary vector contains instructions to check locations 1 and 2, the server places a status poll call to the status poll VDN at the New Jersey and Denver servers and compares the results. If location 2 is better than either location 1 or any splits that are checked on the originating server, the call is interflowed to the interflow VDN that is specified in the plan for location 2.

Simple multisite BSR

Multisite BSR compares local and remote splits or skills and queues calls to the resource that provides the best service. Remember that each BSR application has the following main parts:

- An application plan that identifies the remote servers for comparison.
- A set of the following VDN or vector pairs:
  - Primary: The primary vector, which is on the origin server, contacts the specified remote servers, collects information, compares the information, and routes the call to the server that is likely to provide the best service.
  - Status poll: The status poll vector, which is on the remote server, compares resources on the remote server and replies to the origin server with information on the best of the resource. Each remote server in a given application must have a dedicated status poll vector.
  - Interflow: When a given remote server is the best available, the origin server interflows the call to the vector on the remote server. Each remote server in a given application must have a dedicated interflow vector.

The general operational scheme for multisite BSR is shown in the following figure.



The following example shows the primary VDN using a multisite BSR application.

BSR example primary VDN

```
change vdn xxxxx                                                page 1 of 3
        VECTOR DIRECTORY NUMBER

                            Extension: 52222
                                 Name: Multisite BSR
                        Vector Number: 222
                  Attendant Vectoring? n
                   Meet-me Conference? n
                   Allow VDN Override? n
                                  COR: 59
                                   TN: 1
                             Measured: internal
       Acceptable Service Level (sec): 20
              Service Objective (sec):
        VDN of Origin Annc. Extension:
                            1st Skill:
                            2nd Skill:
                            3rd Skill:
```

```
change vdn xxxxx                                                page 2 of 3
VECTOR DIRECTORY NUMBER
                           Audix Name:
                   Return Destination:
               VDN Timed ACW Interval:
                      BSR Application: 15
         BSR Available Agent Strategy: UCD-MIA
```

In the example shown above for VDN 52222, the entry in the **BSR Application** field links this VDN to BSR Application Plan 15. Also note the UCD-MIA entry in the **BSR Available Agent Strategy** field. If vector 222 uses BSR commands, calls are not automatically delivered to the first resource found with an available agent. All **`consider`** steps in vector 222 are executed, and one of the following happens:

If	Then
There is no available agent in the local or the remote splits	The call queues to the split with the lowest adjusted EWT.
Only one split has an available agent	The call is delivered to the split.
More than two splits have available agents	The call is delivered to the split with the most idle agent.

Note that the **Allow VDN Override** field is set to `n`. If a second VDN and vector are used to process the call, the UCD-MIA strategy and the application plan specified in VDN 52222 are used.

Application plan 15, which is shown in the sample multisite BSR application plan, identifies the remote server and provides the digit strings to dial into the VDNs for both the status poll vector and the interflow vector.

BSR primary vector

When a call arrives at the origin server, the primary vector processes the call. The vector begins the BSR process by checking the specified resources. The following example shows a primary vector used for that purpose.

BSR example of primary vector on origin Communication Manager

```
1. wait time 0 secs hearing ringback
2. consider split   1 pri m   adjust-by 0
3. consider location 2 adjust-by 30
4. queue-to-best
```

In this example, the **`consider`** commands in steps 2 and 3 collect information to compare local split 1 with more than one split at location 2. Location 2 is the Denver server identified on the BSR Application Plan screen. Step 4 queues the call to the best found split. As in singlesite BSR, with the adjust-by parameter of the **`consider`** command, you can set preferences for each resource, whether the resource is a remote location or a split/skill on the origin server. In multisite BSR, with the user adjustment, you can control the frequency of interflows by adjusting the EWT that is returned by a particular resource on a remote server. In this example, the Communication Manager administrator has chosen to adjust the EWT value for location 2 by 30.

BSR status poll vector

To collect information from the remote server, the **`consider location 2 adjust-by 30`** command in the primary vector places an ISDN call, known as a status poll, to the status poll vector on the server at location 2. The following example shows a status poll vector on the remote server used for the purpose.

BSR example of status poll vector on remote Communication Manager

```
1. consider split  2    pri m   adjust-by 0
2. consider split 11    pri m   adjust-by 0
3. reply-best
```

The status poll retrieves and returns information to the origin server. The call is not connected to the status poll VDN.

The vector compares splits 2 and 11, identifies the best split and sends the information back to server 1 with the **`reply-best`** command. Note that the **`adjust-by`** command can be used on the remote server to adjust the EWT returned by either of the splits. When EWT adjustments are applied at both the origin and remote servers, the two adjustments are added at the origin server.

The **`consider`** command is ISDN neutral and does not return answer supervision. The status poll call drops after vector processing executes the **`reply-best`** step, but the ISDN DISCONNECT message returned to server 1 contains the information from the best split considered at location 2. Once the remote server returns the necessary information, the **`consider`** series in the primary vector on server 1 can continue at the next vector step.

Caution:

Do not use the status poll vectors to poll other servers. Status poll vectors compare resources only on the server where the vector resides. Status poll vectors must always end with a **`reply-best`** command. Do not use a **`busy`** or **`disconnect`** command.

Note:

Multisite BSR includes mechanisms that automatically limit the number of status poll calls that can be placed over the network when such calls are unlikely to yield better service for the caller.

BSR interflow vector

In this example, no agents are available and split 11 (location 2) has the lowest adjusted EWT. The **`queue-to best`** command in the primary vector interflows the call to the interflow vector at location 2. The following example shows what the interflow vector looks like.

BSR example of interflow vector on remote Communication Manager

```
1. consider split   2   pri m   adjust-by 0
2. consider split 11     pri m   adjust-by 0
3. queue-to best
```

The interflow vector checks the status of both splits again to get current information and queues or delivers the call to the best split. Note that the consider sequences in the interflow vector and the status poll vector are identical aside from their last step. When a call is interflowed, the call is removed from any queues at the origin server and any audible feedback at the origin server is terminated.

Caution:

BSR will not operate correctly unless the consider series in the status poll vector and the interflow vector use the same splits or skills with the same queue priorities.

BSR call interflow with SIP

You can use SIP to interflow BSR calls, but not BSR polling. The polling must be done using either H.323 or ISDN trunks. Once polling determines the best site, the incoming call can be routed to another location by the queue-to-best step. The redirected call can be routed over a SIP trunk.

To use BSR local treatment for calls queued remotely, the queued call must be answered before expiration of the Session Establishment Timer for the SIP trunk signaling group. You must increase the Session Establishment Timer to an appropriate value. For an approach to handling exceeding the timer, see the local treatment “Take back example” in the “Example vector for the local Communication Manager” topic.

Related links

Example vectors for the local Communication Manager on page 95

What happens to the call if the interflow attempt fails?

If the interflow attempt fails, for example, because there are no available trunks, the call is queued to the best local split. The call is not disconnected. The call is not dropped from vector processing on the origin server. For the call to be queued to a local split, however, that split must be checked at some previous point in the consider series. In writing primary vectors, always check local splits or skills before remote resources.

Adjusting the AIT or AOC returned by an available resource

Procedure

1. Enable the **Available Agent Adjustments for BSR** field on the Feature-Related System Parameters screen.
2. On the Vector screen, administer a **`consider split/skill`** or **`consider location`** vector command specifying both the split or skill, or location and the adjust-by parameter.

   You can use the adjust-by parameter to provide a percentage value during vector processing and can be:

   - A percentage (0 through 100)
   - A vector variable (A-Z, AA-ZZ)
   - A VDN variable (V1-V9)

Result

Once the vector command is executed, the adjustment factor has the following result when the remote site has an available agent:

- For the MIA strategies, the adjustment reduces the Agent Idle Time (AIT) received.
- For the LOA strategies, the adjustment increases the Agent Occupancy percentage (AOC) received.

Depending on the available agent strategy assigned to the VDN for the call, the adjusted AIT or the adjusted AOC is used for that local split or skill, or remote location when choosing between available agents over multiple locations.

Example: You have an agent whose current AIT is 40 percent. You can increase the agent idle time to 60 percent to prevent sending calls to the remote location. If the strategy is `ucd-loa`, use the following vector command:

`consider location 4 adjust-by 50`

The occupancy used for location 4 is increased by 50 percent of the actual occupancy. The occupancy originally sent was 40 percent. A 50 percent adjust-by results in addition of 40 to 50 percent of 40. Therefore, 40 + 20 = 60 percent.

BSR available agent strategies

In multisite BSR applications, the *1st-found* available agent strategy results in fewer interflows and minimizes the load on trunking. Communication Manager has less processing to perform for each call in BSR vectors as Communication Manager does not compare many resources to identify the best resource.

If the processing power and tie trunk capacity are issues in your multisite applications, use the 1st-found strategy.

The other strategies typically result in a much greater percentage of calls being interflowed, optimizing load balancing across locations.

More on status poll and interflow vectors

Read the following points before you write status poll and interflow vectors:

- Since status poll vectors do not return answer supervision, call charges are not normally incurred for the status poll portion of the call flow.
- When a **`consider location`** step performs a status poll, the command also checks for the availability of a B-channel. If no B-channel is available, the remote resource is not treated as the best since the call cannot be redirected to a remote resource.
- If only one split or skill on a remote server can service the call type handled in a BSR application, you do not have to write a consider series in the interflow vector. You can just queue the call to the appropriate resource.
- If status poll and interflow vectors check more than one split or skill, the VDNs for the vectors must be administered with the appropriate BSR available agent strategy.

User adjustments in multisite BSR

User adjustments are especially important in multisite applications, where unnecessary interflows are costly and use trunk capacity inefficiently.

User adjustments in multisite applications function in the same way as in singlesite BSR with one important difference: user adjustments is applied at the remote servers in an application as well as at the origin server. Since a status poll vector uses **`consider`** steps to evaluate resources on the server where the vector resides, with the *adjust-by* portion of each **`consider`** command, the administrator at each server can set preferences for the splits or skills at the server. In BSR applications, the status poll vector checks any such adjustment for a split or skill when selecting the best resource. The adjustment is then returned to the origin server along with the other data for that resource. When the server receives the adjustment from the remote server, the server adds the adjustment to any adjustment that was assigned to that location in the **`consider`** location step. In the following example, no agents become available during the time the vectors are processing the call.

The following example shows a primary vector that checks one remote location to which the vector assigns an adjustment of 30.

Vector with consider step for one location

```
1. wait time 0 secs hearing ringback
2. consider split     pri m  adjust-by  0
```

```
3. consider location 2 adjust-by 30
4. queue-to-best
```

The following example shows the status poll vector at location 2.

Status poll vector

```
1. consider split  2    pri m   adjust-by 0
2. consider split 11    pri m   adjust-by 20
3. reply-best
```

`Consider split/skill` commands in the status poll vectors work the same as in singlesite BSR vectors. User adjustments are applied to a single split or skill and not to the entire location. In this case, the two splits are assigned different adjustments. Say that split 11, despite having the larger adjustment, returns the lower adjusted EWT for a call. The **`reply-best`** command in step 3 returns the user adjustment of 20 to the primary vector on the origin server, along with the rest of the data for split 11.

In saving the data that is returned by location 2, the origin server adds the remote adjustment of 20 to the adjustment of 30 that is specified in step 3 of the primary vector. As a result, the call does not interflow to location 2 in this example unless the EWT for location 2 is more than 50 seconds better than the EWT in split 1 on the origin server.

Example of multisite BSR with limited trunking

Multisite BSR applications must balance improvements in wait times and agent utilization with the cost of interflows and the availability of inter-server trunking for status polls and interflows.

The following example shows an application used for balancing agent workload across the network while still limiting tie trunk usage.

BSR example of Application Plan

```
BEST SERVICE ROUTING APPLICATION PLAN

Number: 10   Name: International       Maximum Suppression Time: 60   Lock? y

Num Location Name    Switch Node           Status Poll    VDN   Interflow VDN  Net Redir

1 Kansas City   1111   919131234015        919131234115         n
2 New York      1112   912121234015        912121234115         n
3 Montreal      1113   915141234015        915141234115         n
3 London        1114   90114411234015      90114411234115       n
___ ______________      ____  ____   __________  __      ____________         n
___ ______________      ____  ____   __________  __      ____________         n
___ ______________      ____  ____   __________  __      ____________         n
___ ______________      ____  ____   __________  __      ____________         n
___ ______________      ____  ____   __________  __      ____________         n
___ ______________      ____  ____   __________  __      ____________         n
___ ______________      ____  ____   __________  __      ____________         n
___ ______________      ____  ____   __________  __      ____________         n
___ ______________      ____  ____   __________  __      ____________         n
___ ______________      ____  ____   __________  __      ____________         n
___ ______________      ____  ____   __________  __      ____________         n
```

The following VDN example shows the VDN screen for VDN 51110, the VDN used in the BSR Application Plan example. In the example, the entry in the **BSR Application** field links the VDN to BSR Application Plan 10. Also note the `EAD-MIA` entry in the **BSR Available Agent Strategy** field.

If vector 100 uses BSR commands, calls are not automatically delivered to the first resource found with an available agent. In each consider sequence, when the **`queue-to best`** or **`check best`** step executes, one of the following happens:

Condition	Action
No skill has an available agent	The call queues to the skill with the lowest adjusted EWT.
One skill has an available agent	The call is delivered to that skill.
More than two skills have available agents	The call is delivered to the skill with the most expert agent, which is the agent with the lowest skill level.
More than two skills have available agents with the same skill level	The call is delivered to the skills that has the most idle agent.

Note that the **Allow VDN Override** field is set to `n`. If a second VDN and vector are used to process the call, the EAD-MIA strategy and the application plan specified for VDN 51110 is used.

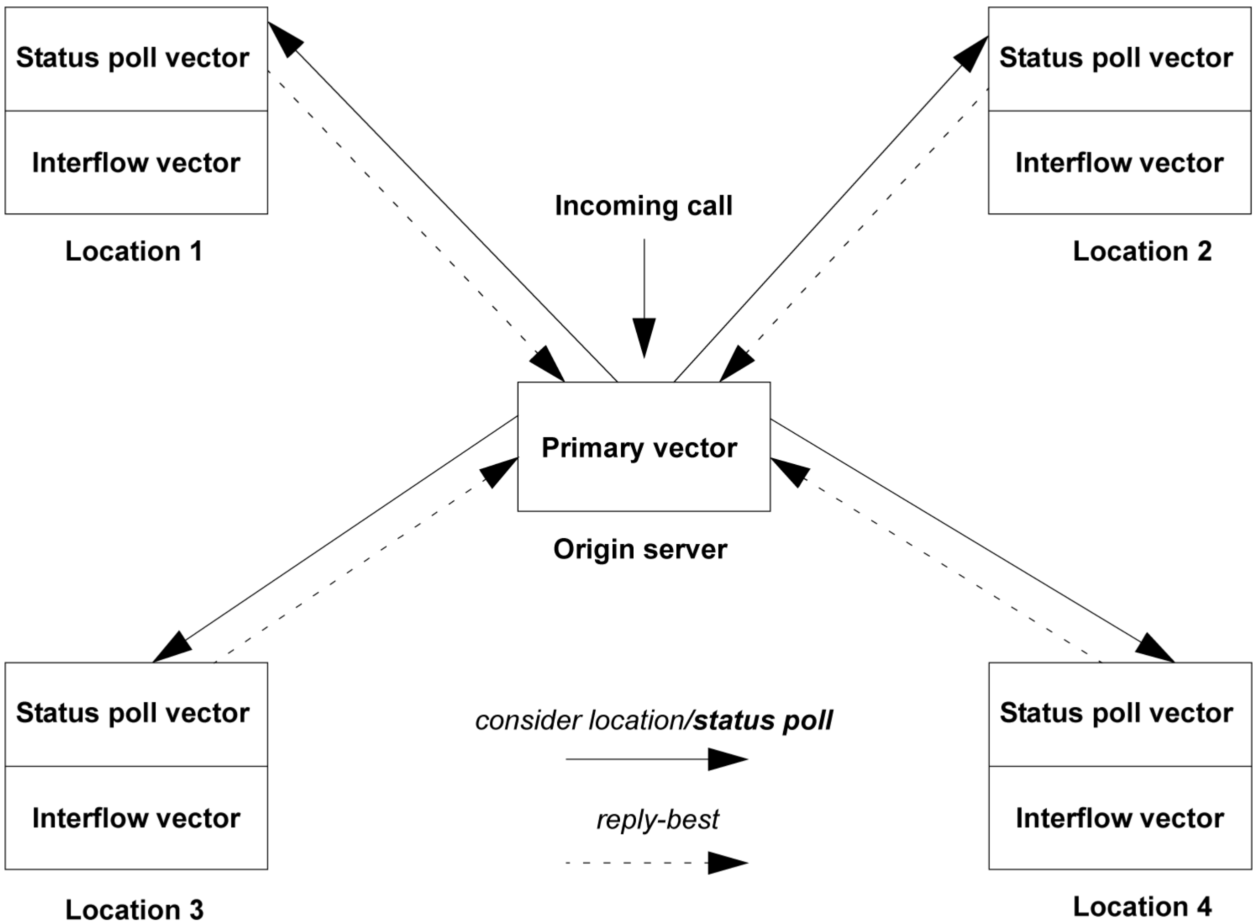
BSR example of primary VDN

```
change vdn xxxxx                                                  page 1 of 3
        VECTOR DIRECTORY NUMBER

                            Extension: 51110
                                 Name: Multisite BSR
                        Vector Number: 100
                  Attendant Vectoring? n
                   Meet-me Conference? n
                   Allow VDN Override? n
                                  COR: 59
                                   TN: 1
                             Measured: none
       Acceptable Service Level (sec): 20
              Service Objective (sec):
        VDN of Origin Annc. Extension: 1001
                            1st Skill:
                            2nd Skill:
                            3rd Skill:
```

```
change vdn xxxxx                                                 page 2 of 3
        VECTOR DIRECTORY NUMBER

                           Audix Name:
                 Messaging Server Name:
                    Return Destination:
                VDN Timed ACW Interval:
                       BSR Application: 15
         BSR Available Agent Strategy: UCD-MIA
             Observe on Agent Answer?: n
```

The overall application is represented in the following figure. Application plan 10 on the origin server identifies the remote servers and provides the digit strings to dial into the VDNs for both the status poll vector and the interflow vector on each server.

Each **`consider location`** command in the primary vector places a status poll call to its specified location. The status poll vector at that location executes a series of **`consider skill`** commands and returns data on the best resource to the origin server through a **`reply-best`** command.



The following example shows the primary vector for this application. The first **consider** series in the primary vector tests two local skills. If either skill has an available agent, step 4 jumps to step 9 and the call queues locally. No remote locations are polled. If no agents are available in either local skill, steps 5 to 8 test four remote locations. In general, do not use other commands between the **consider** steps. The use of the **goto** step is one of the few exceptions to the rule.

If the best remote location's adjusted EWT can reduce the call's current adjusted EWT, step 9 interflows the call to the location. In the vector, a local available agent is always favored over a remote available agent. The call is always directed to the most idle and skilled agent , irrespective of the location that services the call

Multisite BSR example

```
1. wait time 0 secs hearing ringback
2. consider skill 1 pri m   adjust-by  0
3. consider skill 2 pri m   adjust-by 20
4. goto step 9 if expected-wait for best = 0
5. consider location 1      adjust-by 30
6. consider location 2      adjust-by 30
7. consider location 3      adjust-by 50
8. consider location 4      adjust-by 50
9. queue-to best
10. announcement 1001
11. wait time 60 secs hearing music
12. goto step 10 if unconditionally
```

In the primary vector, note that user adjustments are entered for local skill 2 as well as for all the remote locations. The user adjustments indicate the administrator's preferences regarding both local and remote resources. If neither local resource has an available agent, the EWT is greater than 0.

Status poll vector in a multisite BSR application

Each receiving server in a multisite application must have a status poll vector. To collect information from the locations, each **`consider location`** command in the primary vector places a status poll to the status poll vector for the appropriate server.

The following example shows the status poll vector on the server at location 3.

BSR example of status poll vector at location 3

```
1. consider skill  2   pri m  adjust-by  0
2. consider skill 11   pri m  adjust-by 20
3. consider skill 21   pri m  adjust-by 30
4. reply-best
```

The status poll vector compares skills 2, 11, and 21, identifies the best skill, and sends information back to the origin server through the **`reply-best`** command. Note that user adjustments are applied to skills 11 and 21 to adjust the EWT. When EWT adjustments are applied at both the origin and remote servers, both the adjustments are added at the origin server.

If skill 11 has the best adjusted EWT at location 3, the skill data, including a user adjustment of 20, is returned to the origin server by the **`reply-best`** command.

Finding the best resource

Once the remote servers have returned the best data for each location, the second consider series in the primary vector can be completed. In this example, let's suppose that no agents are available at any remote location.

The following table shows how user adjustments at the origin and remote servers yield the adjusted EWT for each location.

Location	Actual EWT of remote best (sec)	User adjustment on origin server	User adjustment on remote server	Adjustment applied by origin server (sec)	Adjusted EWT used in BSR calculations (sec)
1	60	30	0	30	90
2	45	30	10	40	85
3	40	50	20	70	110
4	70	50	0	50	120

The second consider series identifies location 2 as the best remote location, with an adjusted EWT of 85, and the queue-to best step interflows this call to location 2.

Interflow vector in a multisite BSR application

The interflow vector on a remote server in a multisite application accepts the interflowed call from the origin server. The vector also executes the consider series as the status poll vector to identify the current best resource, in case conditions have changed since the status poll.

The following example shows the interflow vector on a remote server.

BSR example of interflow vector at location 2

```
1. consider skill  2     pri m   adjust-by 0
2. consider skill 11     pri m   adjust-by 20
3. consider skill 21     pri m   adjust-by 30
4. queue-to best
```

When a call is interflowed, the call is removed from any queues at the origin server and any audible feedback at the origin server is terminated.

Caution:

BSR does not operate correctly unless the consider series in the status poll vector and the interflow vector use the same splits/skills with the same queue priorities.

Example of multisite BSR with slow networks

Network response times are not an issue for most users. This example is intended for those users, if any, who experience such a problem. This example uses the same VDN, application plan, and a four-server network that is described in the example of multisite BSR with limited trunking. The vector in the example minimized interflows by using a **`goto`** step that skips the remote **`consider`** series if a local resource has an available agent. The design is especially useful if network response times are slow. Calls are always queued locally once before being queued at remote locations.

Both status polls and interflows are conditional. The call can wait in the queue for a local resource while BSR looks for a better split or skill at remote locations.

The example also shows the function of the **`check best`** command and the wait-improved conditional.

The following example illustrates the primary vector for the application, vector 100. The first consider series in the primary vector tests two local splits and queues the call to the best split. If the EWT for the best split is less than 30 seconds, step 5 jumps to the loop in step 11 and the second consider series is not executed. If the EWT for the best split is more than 30 seconds, steps 6 through 9 test 4 remote locations. If the best remote location reduces the EWT of a call by more than 30 seconds as compared to its EWT in the best local queue, step 10 interflows the call to the location.

Caution:

Queue calls once before using the wait-improved conditional in a vector step. If calls are not already queued when the step with the wait-improved conditional executes, the server reads the EWT of the call as infinite. This can result in a vector that interflows all calls, even if interflowing not the intended function.

Multisite BSR with EWT

```
1. wait time 0 secs hearing ringback
2. consider skill   1  pri m   adjust-by  0
3. consider skill   2  pri m   adjust-by 20
4. queue-to-best
5. goto step 11 if expected-wait for call <= 30
6. consider location 1         adjust-by 30
```

```
7. consider location 2         adjust-by 30
8. consider location 3         adjust-by 50
9. consider location 4         adjust-by 50
10. check best if wait-improved > 30
11. announcement 1001
12. wait time 60 secs hearing music
13. goto step 11 if unconditionally
```

A **consider** series can end with either a **queue-to best** or a **check best** step. The **check best** command lets you set conditions that must be met before a call is queued to the best resource. In this example, step 10 in the primary vector is **check best if wait-improved > 30**. In other words, step 10 interflows the call to the best location found by the **consider** series only if the EWT for the location is more than 30 seconds better than the EWT of the call in the local queue.

You can use up to 3 **consider** series in one vector. You can write more than 3 **consider** series in a vector, but there is no benefit in doing so. The server only allows you to queue a call simultaneously to 3 different local resources. Since each **consider** series ends by queuing a call, if no agents are available, using more than 3 series in a vector does not place the calls in additional local queues. If the call interflows to another Communication Manager, the call is removed from vector processing and any queues the call was in on the origin server.

You can combine singlesite and multisite **consider** series, as this example shows. Note that user adjustments are entered for local skill 2 as well as for locations 3 and 4. These indicate your preferences regarding both local and remote resources. In this example, say that step 2 queues the call to skill 1, which has an EWT of 65 seconds, before the second **consider** series is executed.

Status poll vector in a multisite BSR with slow networks

Each receiving server in a multisite application must have a status poll vector. To collect information from the locations, each **consider location** command in the primary vector places a status poll to the status poll vector for the appropriate server.

The following example shows the status poll vector on the server at location 3.

BSR example of status poll vector at location 3

```
1. consider skill  2  pri m  adjust-by  0
2. consider skill 11  pri m  adjust-by 20
3. consider skill 21  pri m  adjust-by 30
4. reply-best
```

The status poll vector compares skills 2, 11, and 21, identifies the best skill, and sends information back to the origin server through the **reply-best** command. Note that user adjustments are applied to skills 11 and 21 to adjust the EWT. When EWT adjustments are applied at both the origin and remote servers, both adjustments are added at the origin server.

If skill 11 has the best adjusted EWT at location 3, the skill data, including a user adjustment of 20, is returned to the origin server by the **reply-best** command.

The first **consider** series queues the call to local skill 1. If the second **consider** series identifies location 2 as the best remote resource, the **check** command in step 10 recalculates the current and unadjusted EWT of the call in skill 1. The **check** command compares the calculated value to the unadjusted EWT of location 2. If the call’s actual (unadjusted) EWT can be improved by more than 30 seconds, the call is interflowed.

Note:

BSR uses adjusted EWT to determine the best resources in the **`consider`** series. Once the best resource is identified, subsequent **`expected-wait`** and **`wait-improved`** conditionals use the actual EWT values.

Interflow vector in a multisite BSR with slow networks

When a call is interflowed to any of the remote locations, the interflow vector on that server accepts the interflowed call from the origin server. It also executes the same consider series as the status poll vector to identify the current best resource, in case conditions have changed since the status poll. The following example shows such an interflow vector.

BSR example of interflow vector at location 2

```
1. consider skill 2   pri m  adjust-by  0
2. consider skill 11  pri m  adjust-by 20
3. consider skill 21  pri m  adjust-by 30
4. reply-best
```

Caution:

BSR will not operate correctly unless the consider series in the status poll vector and the interflow vector use the same splits/skills with the same queue priorities.

If the call is queued to a remote resource by step 10 in the primary vector, is the call removed from the local queue that it entered in step 4? When a call is interflowed, the call is removed from any queues at the origin server and any audible feedback at the origin server is terminated.

The second consider series can compare local and remote resources. If it does, and if step 10 queues the call to another local skill, will the call be removed from the local queue that it entered in step 4?

No. In general, the server can queue a call to as many as 3 local splits or skills simultaneously. BSR does not change this limit.

Example for handling excessive wait times

Step 4 of the vector directs calls to a **`disconnect`** step after an **`announcement`** step when wait time in the network exceeds five minutes.

The following example shows a simple primary vector.

Multisite BSR using disconnect for excessive wait times

```
1. wait 0
2. consider skill    1   pri m   adjust-by 0
3. consider location 2   pri m   adjust-by 30
4. goto step 6 if expected-wait for best ≤ 300
5. disconnect after announcement 3001
6. queue-to best
```

Advanced multisite routing

This section is intended for users whose call center networks meet the following criteria:

- More than five switches in the network.
- Combination of low-volume and high-volume locations.

Application architecture in multisite BSR

Multisite applications can be structured in a variety of ways. In general, however, most applications fit one of two models: distributed or centralized. When each switch in a network interflow calls to other switches and receive interflows, this is called a distributed system. A centralized system, by contrast, is one in which all calls are initially delivered to a single call center, the hub, and distributed from this site to queues at remote switches. A centralized system requires greater inter-switch trunking, since a greater percentage of calls must be redirected. However, if your organization has a significant investment in VRU and CTI technology at the hub, the configuration is appropriate.

Which architecture you choose for an application has direct implications for your choice of user adjustments and polling patterns.

User adjustment considerations

User adjustments in **`consider split/skill`** steps can be set at the user discretion. In distributed multisite applications, however, pay attention to adjustments due to costs and inter-switch trunk capacity. In centralized applications, all calls are redirected so you can use adjustments of zero (0). In distributed applications, a user adjustment of zero (0) for a **`consider location`** step is neither practical nor efficient.

In distributed applications, if the adjustments are small, the load balance across the network is less, but the percentage of calls redirected between switches is high increasing the demands on inter-switch trunking. Higher adjustments reduce interflows, but at the cost of allowing greater imbalance in the load between switches. It takes time and effort to find the best combination of user adjustments in a particular network, but the table contains recommended ranges for initial user adjustments under different conditions. Adjustments can vary between different call center applications so apply the guidelines for each of your applications separately.

Recommended adjustments	Criteria
10-15	• To balance wait times across the network. • Trunk facilities between Communication Manager are plentiful. • Each Communication Manager receives more than 1 call every 10-15 seconds, that is, more than 240-360 calls per hour, for the application.
20	• To balance wait times across the network. • Adequate trunk facilities are available to support the desired balance. • Each Communication Manager receives more than 1 call every 20 seconds, that is, more than 180 calls per hour, for the application.

Table continues…

Recommended adjustments	Criteria
More than 30	• Gains in agent efficiency are more important to you than balancing wait times across the network. • Trunk facilities are scarce. • Call interflow is costly. • Each Communication Manager receives no more than 1 call every 30 seconds, that is, less than 120 calls per hour, for the application.

In the first multisite application, you can begin with a remote adjustment of 30. You can reduce the adjustment later if inter-switch trunking is under-utilized. On the other hand, if trunk exhaustion is a common occurrence, user adjustments are probably set too low. When trunks are exhausted, no further load balancing takes place, deteriorating the overall balance.

Set high user adjustments so that calls are not interflowed to gain the equivalent of a fraction of a queue position. The following equation gives the minimum recommended user adjustment for each remote switch:

$$\frac{\text{AverageCallHandlingTime}}{\text{NumberOfFullTimeEquivalentAgents}} \leq \text{UserAdjustment}$$

Adjustments for remote locations is probably in the range of 10–30 in most distributed applications.

User adjustments and the balance in wait times

Changing conditions can produce significant variations between user adjustments and the balance in wait times across a network, but on average you can predict the balance in wait times for a given user adjustment.

Choose, for instance, a user adjustment of 20 for all remote resources in a network and poll all the remote sites. When waiting time is short, less than 100 seconds, the highest and lowest EWTs for the application on the network stays within a range of approximately 20 seconds. When waiting time is long, more than 100 seconds, the highest and lowest EWTs for the application stays within a range of approximately 20 percent.

Status polling

Status polls are the key element in multisite BSR applications. Status polls provide the communication link between Communication Manager that interflows a call and other Communication Manager that provides service to the call.

The vectors that you write in multisite applications must balance the costs of time and trunk usage with the benefit of better customer service. BSR is designed to help you achieve this balance, incorporating mechanisms to maximize improvements in customer service while minimizing interswitch communications with attendant delays and trunk usage. This section explains the mechanisms and the benefits.

How long do status polls take?

One **consider location** step polls one remote location. Does this mean that an optimal multisite BSR application polls every switch in a network? No.

Let us look at an example of a moderately large network, containing 16 switches. The primary vector on switch 1 can be written as shown in the following vector example. Polling response times are variable. If this is a slow response network and each status poll takes 1 second, the **consider** series in this vector adds as much as 15 seconds to a time of a call in vector processing. In fact, the vector in the example of what NOT to do.

Intelligent polling for multi switch networks

```
1. wait time 0 secs hearing ringback
2. consider skill     1 pri m  adjust-by  0
3. consider skill     2 pri m  adjust-by 20
4. goto step 20 if expected-wait for best = 0
5. consider location 1         adjust-by 30
6. consider location 2         adjust-by 30
7. consider location 3         adjust-by 30
8. consider location 4         adjust-by 30
9. consider location 5         adjust-by 30
10. consider location 6        adjust-by 30
11. consider location 7        adjust-by 30
12. consider location 8        adjust-by 30
13. consider location 9        adjust-by 30
14. consider location 10       adjust-by 30
15. consider location 11       adjust-by 30
16. consider location 12       adjust-by 30
17. consider location 13       adjust-by 30
18. consider location 14       adjust-by 30
19. consider location 15       adjust-by 30
20. queue-to best
21. announcement 1001
22. wait time 60 secs hearing music
23. goto step 21 if unconditionally
```

First, even in very large networks you can obtain nearly all of the possible benefits in agent utilization with very few polling connections. In a network of 16 switches, 99 percent of the total benefits possible with BSR can be obtained if each switch polls just 4 others.

Now our vector looks like the following. Is polling time now cut from 15 seconds to 4 seconds, proportional to the reduction in consider steps?

```
1. wait time 0 secs hearing ringback
2. consider skill     1 pri m   adjust-by  0
3. consider skill     2 pri m   adjust-by  0
4. goto step 9 if expected-wait for call = 0
5. consider location 5          adjust-by 30
6. consider location 10         adjust-by 30
7. consider location 13         adjust-by 30
8. consider location 15         adjust-by 30
9. queue-to best
10. announcement 1001
11. wait time 60 secs hearing music
12. goto step 10 if unconditionally
```

In fact, polling time in this vector can be around 0.4 seconds per call because of mechanisms in BSR that constantly react to network conditions and resource usage to minimize the number of status polls. The mechanisms, whose combined operation is called “Intelligent Polling”, also function to make each status poll as productive as possible.

Intelligent polling

A BSR application only polls the switches that are likely to provide the best service at any given time. If a remote switch is polled and returns an adjusted EWT greater than that of the current best resource, polling of the remote switch is suppressed for a period of time proportional to the difference between the two adjusted EWT values. In other words, polling of a given location is suppressed whenever the adjusted EWT returned by that location is subsequently replaced by a better adjusted EWT from another resource. The **`consider`** step for this location is skipped during the period and vector processing continues at the next step. When the suppression period is over, the **`consider`** step once again polls the location. If the location returns the best adjusted EWT, the next call processed by the vector also causes the location to be polled. If the location is not the best, polling is temporarily suppressed.

If no calls are in queue at the remote location an agent can become available at any moment, and BSR, therefore, does not suppress polling for longer than 5 seconds in such situations. BSR does not suppress polling of any remote location for more than 60 seconds, regardless of the differences between adjusted EWT returned by different switches.

Other conditions can also suppress status polls to a location:

- Resource exhaustion, that is, no trunks available, queue full.
- Administration errors, that is, badly written vectors, or no application plan.

This feature significantly reduces the average number of status polls placed per call. The greater the call volume, the greater the percentage reduction. Let's take another look at the vector in screen 2.

Let us say the network is operating in a balanced state. EWTs are 30 seconds at all locations and a call arrives every 3 seconds at each site. Adjusted EWTs are 30 seconds at the origin switch and 60 seconds for each remote switch. After each status poll under the conditions, polling is suppressed for 30 seconds. Each remote location is polled, therefore, by every 10th call. On average, this means that each call polls any one location 0.1 times. Since there are four **`consider`** steps, each call makes 0.4 polls. Remembering the 1-second polling response time given at the beginning of the example, the average time added to call processing for each call is 0.4 seconds.

The 1st-found available agent strategy, discussed in BSR can cut average polling times further. With the 1st-found strategy, BSR skips all subsequent **`consider`** steps in a series if a resource with an available agent is found and deliver the call to that resource.

Efficient polling patterns in large networks

Unless you have a small network, you cannot benefit by having every switch poll every other switch. This section explains how many remote locations each switch polls and provides guidelines for selecting which locations any given switch must poll.

How many switches must one switch poll?

You do not have to poll every Communication Manager in larger networks. Because of the intelligent polling capabilities of BSR, you can obtain 99 percent of the possible benefits in agent utilization with very few polling connections.

The following example is a laboratory network of 16 Communication Manager used for simulations of BSR multisite applications.

As shown in the following table, approximately 99 percent of the possible benefits are obtained when any one Communication Manager polled four others.

Number of remote sites polled by each switch	ASA across the network (seconds)	Approximate percentage of total benefits obtained (%)
0	192.8	0
1	26.2	89
2	10.6	95
3	7.6	98
4	6.5	99
15	4.7	100

For each Communication Manager to poll the other 11 Communication Manager in the network produces an additional percent gain in ASA and agent utilization, an improvement which is more than offset by the cost of additional messaging and trunking.

In most situations, you can get the optimal results with your multisite BSR applications if you follow the polling guidelines shown in the following table.

Number of switches in the network	Number of switches to poll
2-4	all the other switches
5-10	3 other switches
11-20	4 other switches
21-40	5 other switches
More than 41	6 other switches

Which remote switches must each switch poll?

In networks with less than five switches, each switch can productively poll all the other switches in the network. In larger networks, each switch does not poll every other switch. But which switches must each switch poll? The term *polling patterns* describes the relationships between switches in multisite BSR applications.

Do not use the following polling patterns:

- Mutual polling: As much as possible, 2 switches must not poll each other. This is difficult in small networks, but in large networks mutual polling can and must be minimized.
- Polling chains: For example, if switch A polls B and C, B polls C and D, this is a polling chain.

You can experiment with polling patterns appropriate for your network and applications, if you are not constrained by the physical structure of the network. The following table provides a template for creating polling patterns for applications of up to 12 switches. In the majority of situations, the patterns produce results that are close to optimal. To use the following table, first assign a number from 1 to x to each switch in your application. Next, find the column that matches the number of switches in your application. As you read the column, see which switches each switch in the application must poll.

Switch	Must poll the specific switches shown in the column for your network size							
	5	**6**	**7**	**8**	**9**	**10**	**11**	**12**
1	2,4,5	2,4,5	2,4,6	2,4,7	2,4,6	2,4,7	2,4,8,10	2,4,8,9
2	3,5,1	3,5,6	3,5,7	3,5,8	3,5,7	3,5,8	3,5,9,11	3,5,9,10
3	4,1,2	4,6,1	4,6,1	4,6,1	4,6,8	4,6,9	4,6,10,1	4,6,10,11
4	5,2,3	5,1,2	5,7,2	5,7,2	5,7,9	5,7,10	5,7,11,2	5,7,11,12
5	1,3,4	6,2,3	6,1,3	6,8,3	6,8,1	6,8,1	6,8,1,3	6,8,12,1
6		1,3,4	7,2,4	7,1,4	7,9,2	7,9,2	7,9,2,4	7,9,1,2
7			1,3,5	8,2,5	8,1,3	8,10,3	8,10,3,5	8,10,2,3
8				1,3,6	9,2,4	9,1,4	9,11,4,6	9,11,3,4
9					1,3,5	10,2,5	10,1,5,7	10,12,4,5
10						1,3,6	11,2,6,8	11,1,5,6
11							1,3,7,9	12,2,6,7
12								1,3,7,8

In applications of more than 12 switches, the following table provides the formulae you must figure out the optimal polling pattern.

Number of switches in application	Switch to poll
13 or 16	i + 1, i + 3, i + 7, i +11
14 or 19	i + 1, i + 3, i + 7, i + 9
15	i + 1, i + 3, i + 7, i + 10
17 or 20	i + 1, i + 3, i + 7, i + 12
18	i + 1, i + 3, i + 7, i + 13
21-23	i + 1, i + 3, i + 7, i + 15, i + 17
24	i + 1, i + 3, i + 7, i + 15, i + 19
25	i + 1, i + 3, i + 7, i + 15, i + 20

To use one of the formulae, first assign a number from 1 to x to each switch in your application. Then, in the left-hand column of the table, find the number of switches in your application. The corresponding formula in the right-hand column is the one you must use.

In the formulae, i is the number of the switch for which you are calculating a polling pattern. To calculate the polling patterns in an application with 16 switches, the formula to use is: i + 1, i + 3, i + 7, i +11

As shown in the first row of the table. Here are the actual results of this formulae for the first 5 switches in this 16-switch application. The numbers wrap, that is, start over at 1, after you have polled the last switch in the network: switch 5 polls switch 16 as its fourth poll and then the polling pattern for switch 6 has switch 1 in the fourth position.

Switch number	Switch to poll
1	2 ,4, 8, 12
2	3, 5, 9, 13
3	4, 6 ,10, 14
4	5, 7, 11, 15
5	6, 8, 12, 16
6	7, 9, 13, 1
7	8, 10, 14, 2

Minimizing variations in wait time

When a network contains large and very small resources, BSR and LAI can be effectively combined. This section presents two sample vectors. The first example illustrates a primary vector intended for the small resources in a network wherein a call in queue at one call center when an agent is available at another makes no sense. The design reduces wait time variation as well. The second example illustrates the best way to minimize wait times across a network with large resources.

LAI as a backup

If your principal concern is that a call should not wait in queue while an agent is available elsewhere, use BSR at all locations in the network. At smaller locations, write primary vectors that will perform LAI attempts to other preferably larger resources once the call has been queued.

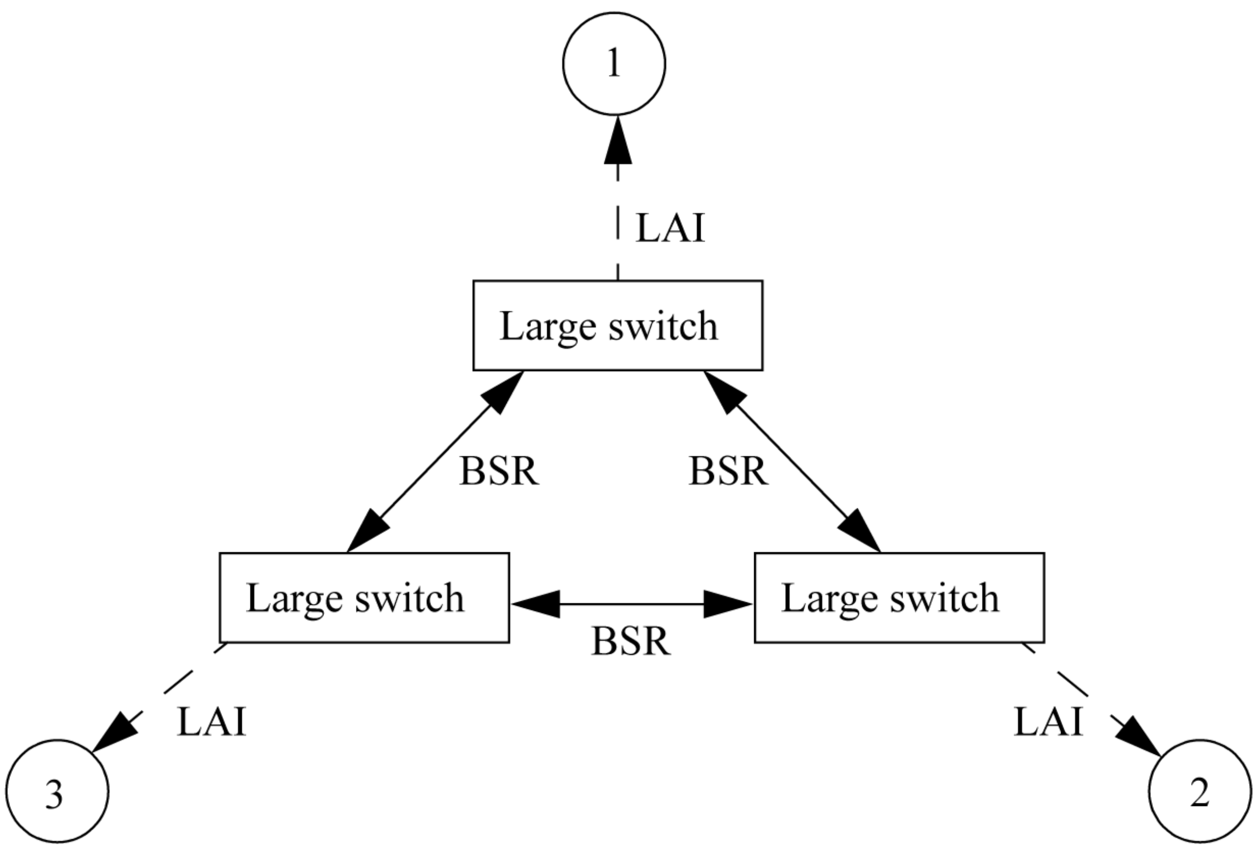
```
1.  wait time 0  secs hearing ringback
2.  consider skill 1st pri m adjust-by  0
3.  consider location 12     adjust-by 30
4.  consider location 22     adjust-by 30
5.  goto step 7 if expected-wait for call < 600
6.  disconnect after announcement 3501 “Due to heavy call volume...”
7.  queue-to skill best
8.  announcement 3500 “Thanks for calling....”
9.  goto step 13 if expected-wait for call < 90
10. wait time 45 secs hearing music
11. announcement 3502 “Still busy...”
12. goto step 9 if unconditionally
13. route-to-number 913031234567 with cov n if interflow-qpos = 1
14. wait time 5 secs hearing music
15. goto step 13 if unconditionally
```

Steps 1 to 4 comprise a typical BSR vector. The origin Communication Manager checks a local resource and 2 remote resources. Before queuing or routing the call, however, the vector checks the expected wait time for the best resource. If this is more than 10 minutes, the caller receives a busy announcement. Otherwise, the queue-to best step sends the call to the best resource. Two vector loops follow: one 45-second loop with music and a delay announcement, and one 5-second loop that uses LAI. If the call is queued successfully in step 7 the first announcement loop (steps 9-12) executes until the call gets within a certain range of the head of the queue (at which point EWT is less than 90 seconds). At this time, step 9 sends the call to the second loop, where LAI attempts are placed every 5 seconds for the call at the head of the interflow eligible queue (**interflow-qpos=1**). If an agent becomes available at the larger remote resource, any call at the head of the eligible queue at the smaller location is outflowed to the larger resource, normally within a period of 5 seconds.

Single-queue FIFO hybrid configuration

To minimize variations in wait time across a network, the best strategy is to let only the call centers with the larger resources receive calls. The following figure shows a network of three large and three small resources, that is, call centers with large skills and call centers very small skills in the same application.

The large locations use BSR and all poll each other, while each location with a small resource (numbered 1, 2, 3) is treated as a satellite of one of the larger locations and receives calls interflowed from the larger location. Mutual polling is not optimal in larger networks, but work well for switches in such a small network to poll each other. BSR is used to balance the load between the locations with the larger resources. Each large switch executes a rapid Look-Ahead Interflow (LAI) vector loop to one small switch to look for available agents. Since calls never queue at the small switches, the problem of highly variable wait times at the small resources is eliminated. This strategy will also give the best balance in wait times across resources.



The following vector example shows the primary vector that is used at the large locations with this strategy. This vector is almost identical to the vector shown in *using LAI as a backup*. The difference is at the application level. In contrast to the previous example:

- Only locations with the larger resources receive calls.

- The primary vector in this example resides on larger switches.

Steps 1 to 4 comprise a typical BSR vector. The origin Communication Manager checks a local resource and 2 remote resources. Before queuing or routing the call, however, the vector checks the expected wait time for the best resource. If this is more than 10 minutes, the caller receives a busy announcement. Otherwise, the queue-to best step sends the call to the best resource. Two vector loops follow: one 45-second loop with music and a delay announcement, and one 5-second loop that uses LAI. If the call is queued successfully in step 7, the first announcement loop (steps 9-12) executes until the call gets within a certain range of the head of the queue. At this time, step 9 sends the call to the second loop, where LAI attempts are placed every 5 seconds (only for the call at the head of the interflow eligible queue). If an agent becomes available at the smaller resource, any call at the head of the eligible queue at the larger location is outflowed to the smaller resource, normally within a period of 5 seconds.

Vector combining BSR and LAI

```
1.  wait time 0  secs hearing ringback
2.  consider skill 1st pri m adjust-by  0
3.  consider location 120    adjust-by 30
4.  consider location 220    adjust-by 30
5.  goto step 7 if expected-wait for best < 600
6.  disconnect after announcement 3501 "Due to heavy call volume..."
7.  queue-to skill best
8.  announcement 3500 "Thanks for calling...."
9.  goto step 13 if expected-wait for call < 90
10. wait time 45 secs hearing music
11. announcement 3502 "Still busy..."
12. goto step 9 if unconditionally
13. route-to-number 913031234567 with cov n if interflow-qpos = 1
14. wait time 5 secs hearing music
15. goto step 13 if unconditionally
```

Similar vector loops can be added to the interflow vectors at each of the large switches. In other words, each vector that processes calls at the larger locations can use rapid LAI loops to interflow calls to its satellite resource. This system maximizes agent utilization and the distribution of call load while evening out wait times across the network.

Low volume splits or skills

Very small resources, for example, 2-3 agents, have special needs. With BSR, you can maintain a balance of wait times across a network of call centers. However, for very small splits or skills, wait times for each call can vary significantly.

To see why this is the case, let us take an extreme example of a split with a single agent logged in with one call active and no call in the queue. Average call handling time is 3 minutes. Now, if a new call arrives in the queue, that call can be answered almost immediately or the call can be in the queue for more than 3 minutes. The variation in wait times is perhaps 5-180 seconds.

In general, the fewer agents logged in to a split or skill, the greater the variability in wait times because agents become available less often. BSR favors large resources, steering calls away from smaller resources when there are no available agents or wait times are not the best in the application. This tendency helps reduce the possibility that an individual caller can have a disproportionately long wait at a small resource.

If your network includes very small splits or skills, you have three options:

- If the operation is not badly affected by a small percentage of calls having variable wait times, use BSR across the network.
- If your principal concern is that a call does not wait in a queue while an agent is available elsewhere, use BSR, but write primary vectors at smaller locations to perform rapid look-ahead attempts to other resources once the call is queued. Rapid LAI vector loops use the **`interflow-qpos`** conditional, which is an enhancement to LAI.
- If you want agents to answer each call quickly, use the following configuration: Do not deliver or queue calls directly to the very small resources. Deliver or queue all incoming calls to larger resources and use BSR to balance the load across the larger locations. Ensure most of the larger locations perform rapid look-ahead attempts to more than one of the smaller resources. In this way, the members of the very small resource become an extension of the agent pool at one of the larger call centers.

In any network, do not have several large resources poll or make look-ahead attempts to a very small resource. As the status at the very small resource changes infrequently, frequent polls to that resource is a waste. A very small resource must receive look-ahead attempts or be polled only by other small resources or by one large resource.

BSR local treatment for calls queued remotely

As the **`converse-on`** vector step does not include any information on the queue position and the wait time for the call at the remote end, Voice Portal Call Back Assist does not work with BSR Local Treatment.

In a multisite BSR configuration, a call that arrives at a local Communication Manager server can be rerouted to a remote server located in a another part of the world. With BSR local treatment for calls queued remotely over IP (SIP or H.323) or ISDN trunks, you can provide local audio feedback to calls while calls wait in a queue on a remote server.

The feature provides the following benefits:

- Bandwidth savings for IP calls.
- No audio quality concerns that occur when music is sent over WAN that use low bit-rate codecs.
- Centralized management for announcements and other call treatment.

Local treatment operations

This section describes local treatment feature operations that occur when BSR redirects IP or ISDN calls to a remote queue.

Important:

Before following the local treatment operations, ensure that the required feature and vector administration steps are implemented on the local *and* remote Communication Manager servers.

The following steps describe the basic process for local treatment operations in a multisite BSR environment:

1. A call arrives at the local Communication Manager server and is processed by a VDN with BSR local treatment.
2. The local vector includes the **`consider`**, **`queue-to best`**, and **`wait`** hearing announcement steps that are required for BSR local treatment operations.
3. A skill on a remote server is identified as the best location and the local server attempts an interflow to the remote server. Vector processing is temporarily suspended on the local server while the interflow attempt is in progress.
4. If the interflow attempt succeeds, the remote server returns an ISDN_PROGRESS message with progress indicator of in-band information (8) or a SIP 182 QUEUED message, to indicate that the call is in a queue and local treatment operations can proceed.

   The remote server must meet the following requirements for the appropriate ISDN_PROGRESS message to be sent back to the local server:

   - The remote server is administered for BSR local treatment.
   - The call is directed to a VDN with BSR local treatment.
   - The vector associated with the VDN includes only those steps and commands that are required for successful local treatment operations.
5. The local server receives a SIP 182 QUEUED or an ISDN_PROGRESS message with progress indicator of in-band information (8), vector processing resumes with an appropriate treatment step and the caller receives feedback provided by the local server while the caller waits in the remote queue.

   Important:

   To ensure that the local treatment feature operates as designed, use the vector commands that are recommended for local treatment implementation. Although local treatment operations do not impose restrictions on the types of vector steps that are administered on the local server after call processing resumes, use of inappropriate vector steps can interfere with local treatment operations.
6. When an ACD agent on the remote server accepts the call, a 200 OK or an ISDN_ALERTING message is sent to the local server. Vector processing is discontinued on the local server and the remote server.

Local treatment system requirements

The BSR Local Treatment for Calls Queued Remotely Over IP or ISDN Trunks feature works on all platforms and operating systems that are supported by Avaya Communication Manager. To use the local treatment feature, the system license file must be configured to enable the following features:

- LAI
- BSR

- BSR Local Treatment for IP and ISDN

Implementation guidelines for local treatment vectors

Important:

Read the guidelines before you implement the local treatment feature.

Implementation of the local treatment feature requires use of specific vector steps to generate the correct ISDN or SIP messages between the local and remote Communication Manager.

For polling vectors: You must be careful to administer your local treatment polling vectors so that calls are not unintentionally dropped or phantom calls are generated. If the **`queue-to best`** step is followed by vector steps that include any commands other than **`announcement`**, **`wait`**, or **`goto`**, the trunk to the remote queue are dropped. For example, the addition of **`consider`** steps after a **`queue-to best`** command can cause intermittent call behavior. The addition of a **`queue-to`** step after a **`queue-to best`** step can cause phantom calls to be queued to the remote server.

Tip:

You can also exploit this functionality to allow the local server to “take back” calls that remain in queue on a remote server after a specified time limit is exceeded. For more information, see the “Take back example” in the example vector for local Communication Manager.

Interflow local treatment vectors on the remote Communication Manager: When the BSR Local Treatment feature is enabled, specific ISDN or SIP messages must be exchanged between the remote and local Communication Manager. If additional vector steps are included either before or after the **`consider`** steps and **`queue-to best`** in the interflow vector on the remote server, the following results occur:

- Either an ALERTING or PROGRESS message, with in-band information, is returned from the remote server to the local server.
- In response to the message, trunk bandwidth is immediately allocated and the call is removed from the local queue.
- Local treatment operations cease, trunk bearer resources are allocated for the call sooner than required and cost savings associated with the local treatment feature are not realized.

Example vectors for the local treatment feature

This section provides vector guidelines and examples that describe how to implement the local treatment feature. Vector administration typically requires polling vectors on both the local and remote Communication Manager and an interflow vector on the remote server. You must also administer the polling vector on the local server to provide an appropriate local call treatment.

Example vectors for the local Communication Manager

The following examples shows two different vector strategies to implement the local treatment feature on the local server. Vectors created for this purpose are the same as those used in all BSR polling operations, which include a **`consider`** series followed by a **`queue-to best`** step.

Important:

Administer the local treatment polling vectors so that calls are not dropped unintentionally.

After various skills and locations are polled and the call is placed in queue at the identified best location, the local server continues to maintain control of the call until the call is answered by an agent. While the call is in queue, the local server continues to provide additional vector steps to implement the local call treatment.

At a minimum, the local treatment vector must include **`announcement`** and **`wait-time`** steps to provide appropriate feedback to the caller. However, the local treatment vector can be designed to use either a continuous loop or take back strategy. The alternate local call treatment strategies are described in the following sections.

Continuous loop example

The following example shows a vector that provides a sequence of call treatment steps on the local server that proceed in a continuous loop until an agent answers the call at the remote location.

In the following vector example, step 6 places the call in queue at the identified best location. Step 7 provides an appropriate announcement and step 8 provides 10 seconds of music. Step 9 uses an unconditional **`goto`** step to loop call processing back to step 6, where the treatment process continues.

```
change vector 40                                                     Page 1 of 3
                                    CALL VECTOR

  Number: 40                     Name: Local BSR vector
                   Attendant Vectoring? n       Meet-me Conf? n            Lock? n
    Basic? y    EAS? y   G3V4 Enhanced? y      ANI/II-Digits? y   ASAI Routing? y
Prompting? y    LAI? y  G3V4 Adv Route? y       CINFO? n BSR? y       Holidays? y

01 announcement 3000
02 consider skill 4 pri m adjust-by 0
03 consider skill 6 pri m adjust-by 0
04 consider location 1 adjust-by 10
05 consider location 2 adjust-by 10
06 queue-to best
07 announcement 3001
08 wait-time 10 secs hearing music
09 goto step 7 if unconditionally
```

Take back example

The previous example set up the local treatment process as a continuous loop that repeats indefinitely while the call remains in queue at the identified best location. However, you can also design vectors that allow the local server to take back a call after the call remains in queue for a specified time.

In the following vector example, the **`queue-to best`** in step 6 is followed by a series of **`announcement`** and **`wait-time`** commands provided in steps 7 through 12. If the treatment steps complete and the call still remains in the remote queue, vector processing proceeds to step 13, which uses a **`route-to`** command that causes the call to the remote server to be dropped. The **`route-to`** step can be used to provide alternate services for the call.

Note:

When the call to the remote server is dropped, a type 305 vector event is logged.

```
change vector 40                                                          Page 1 of 3
                                    CALL VECTOR

Number: 40                    Name: Local BSR vector
                 Attendant Vectoring? n          Meet-me Conf? n             Lock? n
    Basic? y    EAS? y   G3V4 Enhanced? y      ANI/II-Digits? y   ASAI Routing? y
Prompting? y    LAI? y  G3V4 Adv Route? y       CINFO? n BSR? y       Holidays? y

01 announcement 3000
02 consider skill 4 pri m adjust-by 0
03 consider skill 6 pri m adjust-by 0
04 consider location 1 adjust-by 10
05 consider location 2 adjust-by 10
06 queue-to best
07 announcement 3001
08 wait-time 10 secs hearing music
09 announcement 3001
10 wait-time 10 secs hearing music
11 announcement 3001
12 wait-time 10 secs hearing music
13 route-to number 54010 if unconditionally
```

For another method to take back the call based on the time the call has been in the system, see "VDN type variable" in the *Programming Call Vectoring Features in Avaya Aura® Call Center Elite* document.

Related links

BSR call interflow with SIP on page 74

Example polling vector for the remote Communication Manager

The following example shows a call vector that polls skills on the remote server. This vector does not differ from other typical BSR polling vectors.

```
change vector 31                                                          Page 1 of 3
                                    CALL VECTOR

Number: 31                    Name: Remote BSR poll vector
                 Attendant Vectoring? n          Meet-me Conf? n             Lock? n
    Basic? y    EAS? y   G3V4 Enhanced? y      ANI/II-Digits? y   ASAI Routing? y
Prompting? y    LAI? y  G3V4 Adv Route? y       CINFO? n BSR? y       Holidays? y

01 consider     skill    3   pri m adjust-by 0
02 consider     skill    4   pri m adjust-by 0
03 reply-best
```

Example interflow local treatment vector for the remote Communication Manager

The following example shows a call vector that is used to interflow the call to the remote server while local treatment is provided for the call.

Important:

When the BSR Local Treatment feature is enabled, specific ISDN messages must be exchanged between the remote and local Communication Managers. If additional vector steps are included either before or after the consider steps (if used) and queue-to best in the interflow vector on the remote server, the following results occur:

- Either an ALERTING or PROGRESS message (with in-band information) is returned from the remote server to the local server.
- In response to the message, trunk bandwidth is immediately allocated and the call is removed from the local queue.
- Local treatment operations cease, trunk bearer resources are allocated for the call sooner than required and cost savings associated with the local treatment feature are not realized.

```
change vector 32                                                      Page 1 of 3
                                    CALL VECTOR

Number: 32                    Name: Remote BSR interflow vector
                 Attendant Vectoring? n         Meet-me Conf? n            Lock? n
    Basic? y    EAS? y   G3V4 Enhanced? y      ANI/II-Digits? y   ASAI Routing? y
Prompting? y    LAI? y  G3V4 Adv Route? y       CINFO? n BSR? y       Holidays? y

01 consider     skill    3   pri m adjust-by 0
02 consider     skill    4   pri m adjust-by 0
03 queue-to     best
```

BSR-initiated path replacement for calls in vector processing

You can use QSIG or DCS with Reroute using ISDN SSE for path replacement for calls in a queue. For calls that are waiting in a queue or in vector processing, even if the call is not connected to an answering user, Communication Manager can attempt a path replacement to find the optimum path for the call.

The **`queue-to best`** command is used in BSR to initiate a QSIG path replacement for a call. The following scenario can take place:

If the terminating Communication Manager server receives a *Path Replacement Propose* operation for a call that is in a queue or in vector processing, Communication Manager immediately initiates path replacement using the path replacement extension if the **Path Replace While in Queue/Vectoring** field is `y` and the **Path Replacement Extension** field has a valid entry. The fields are available on the ISDN Parameters page of the Feature-Related System Parameters screen.

Example BSR vector written to trigger path-replacement

The following example shows how a BSR vector can be written to trigger path-replacement at the terminating Communication Manager.

Note:

In order for a path-replacement to be attempted, the incoming and outgoing trunks that are used for the call must be administered with the **Supplementary Service Protocol** field set to `b`.

BSR-initiated path-replacement vector

```
1. wait 0
2. consider skill 1
3. consider skill 5
```

```
4. consider location 10    adjust-by 10
5. consider location 24    adjust-by 20
6. queue-to best
```

At the receiving server, the vector that is executed by the incoming call must be programmed with an **`announcement`**, or **`wait hearing music`** vector command. The use of one of the commands is what makes it possible for path-replacement to take place while the call is in vector processing.

BSR interactions

Agent phone display

If Communication Manager forwards the collected digits with an interflowed call, the forwarded digits are displayed on the phones display of the agent who answers the call.

Look-Ahead Interflow (LAI)

Restrictions and interactions that apply to LAI also apply to BSR status poll and interflow calls.

Basic Call Management System (BCMS)

BCMS does not log LAI attempts. Therefore, BCMS does not log BSR status polls, which are treated as LAI attempts.

Call Vectoring

The following considerations apply to all vectors when you set **BSR** to `y`.

Call Vectoring considerations	
`route-to VDN`	If a call is routed to a new VDN, Communication Manager clears the best resource data defined by a series of consider steps in the previous VDN.
`goto vector`	If a **`goto vector`** command is executed, Communication Manager retains the best resource data defined by a series of consider steps in the original VDN and uses the data in the subsequent vector.
`consider`	• Do not use other commands within a series of **`consider`** steps, since the commands can delay the execution of the series. • Skills used in **`consider`** commands must be vector controlled.
`converse`	Communication Manager passes the collected digits that Communication Manager forwards with the call to the VRU using the digits data passing type.
`best` (keyword)	You can use the **`best`** keyword in the following commands, but only with the conditions listed: • **`goto step`** or **`goto vector`** commands using the expected-wait or wait-improved conditionals. • **`check`** commands using the unconditional, expected-wait, or wait-improved conditionals. You cannot use the **`best`** keyword as a replacement for a split or a skill in the following vector commands: • **`converse-on split/skill`** • **`messaging split/skill`**

Direct Department Calling (DDC)

BSR functions when the splits use DDC call distribution. Once Communication Manager determines the best resource, the actual call distribution follows the DDC setting of the split irrespective of the BSR available agent strategy. You cannot use DDC as a BSR available agent strategy.

Distributed networking using QSIG Manufacturers Specific Information (MSI)

BSR does not function with systems from other vendors, unless the vendor develops a corresponding capability that works with Communication Manager.

Expert Agent Selection (EAS)

EAS must use the EAD-MIA or EAD-LOA available agent strategy. You can use the EAS VDN skills, that is, first, second, or third in the **`consider skill`** commands.

Facility Restriction Levels (FRL)

The FRL applies to status poll and interflow calls in the same way FRL works with the **`route-to number`** command.

Integrated Services Digital Network (ISDN)

BSR and globally supported information transport are fully functional over ISDN-PRI or ISDN-BRI trunking facilities.

Note:

You can set up Asynchronous Transfer Mode (ATM) and IP trunking to emulate ISDN-PRI.

Location Preference Distribution

Use location preference distribution to select an available agent within the call center during the **`consider`** and **`queue-to best`** step operations. You cannot use local preference distribution across system sites. In this case, there is no multisite network region.

Look Ahead Routing (LAR) - BSR incompatibility

LAR and BSR are incompatible. If a trunk is not available at the site being polled, an alternative route using an Alternate Route Selection (ARS) pattern can be used to poll, provided there is a secondary route available that supports transporting shared UUI in the DISCONNECT message. This does not use LAR. If no route is available for polling when a **`consider location`** step is executed, BSR processing handles the situation and after a period of 30 seconds, subsequent calls attempt to poll the location again.

The use of alternative routes for polling only works if there are alternative routes for the interflow path, irrespective of whether LAR or BSR is in use.

Multiskill queuing

A call can be queued up to three times by **`queue-to`** or **`check`** commands in the same vector. One vector can therefore contain up to three series of **`consider`** steps. Each series must be followed by a **`queue-to best`** step. Each **`consider`** series selects the best remote resource from the options you specify and queues the call to the resource.

BSR can queue simultaneously only on the origin Communication Manager. BSR no longer controls a call once BSR queues the call at a remote resource.

Network access

BSR interflow operates over public, private, or virtual private ISDN-BRI and ISDN-PRI networks that meet the criteria explained in network requirements. BSR interflow can also operate over SIP trunks. BSR requires the network support transport of user-to-user data using MSI or UUI as a codeset 0 Information Element (IE), or as shared UUI over SIP trunks. The numbers administered on the BSR Application Plan screen are expected to access VDNs using ISDN, H.323, or SIP trunks, or using the Network Call Redirection (NCR) feature.

Administration or call processing does not prevent access to other types of trunks or to destinations that are not VDNs.

Operating Support System Interface (OSSI)

The new administration commands, conditionals, keywords and screens are available using OSSI.

Path replacement for QSIG/DCS ISDN calls

For calls waiting in queue or in vector processing, even if the call is not connected to an answering user, path replacement can be attempted to find an optimal path for the call. This results in more efficient use of the trunk facilities.

The QSIG ISDN or DCS ISDN trunk path-replacement operation can be triggered for ACD calls by the LAI **`route-to number`** vector step, BSR **`queue-to best`** vector step, and the adjunct routing vector steps.

QSIG

LAI, BSR, and information forwarding function over QSIG trunk facilities if the remote locations are Communication Manager.

Redirection on No Answer (RONA)

Calls redirected to a VDN by RONA can be subsequently processed by BSR or LAI applications. When RONA redirects a call to a VDN, any best resource data defined in a previous vector is cleared.

Service Level Maximizer (SLM)

The following interactions occur between BSR and SLM:

- The SLM algorithm applies only within a particular call center location, not across locations in a multisite configuration.
- Assignment of reserve agents applies only to skills within a local site.
- SLM always selects the agent for an SLM skill at the remote site. BSR uses SLM to determine the best available agent and when to route the call to that skill.
- The best skill selected at a particular site or across sites when due to multiple consider steps is based on an existing BSR operation. In other words, the shortest adjusted EWT or skill as defined by the available agent strategy.
- The selection of the agent and delivery of the call in the best-chosen skill is based on what is assigned to the skill.
- BSR does not override the skill distribution algorithms and pick a reserve agent, unless the skill distribution algorithm selects that agent due to the current conditions at that site.

Service Observing

You can observe a call in BSR or LAI processing as long as the call is still connected through the local communication server. All current restrictions on Service Observing still apply.

Transfer

If a call is transferred to a VDN, any best resource data defined in previous vector processing is cleared. Transferred calls do not forward any information such as previously collected digits and In-VDN time, that is forwarded with interflows.

Trunk Access Code (TAC)

Do not use routing numbers, that is, status poll or interflow, that utilize TACs, since the required in-band outpulsing slows the setup operation significantly.

VDN Override

VDN Override applies to the **BSR Application Number** and the **Available Agent Strategy** fields assigned on the VDN screen. VDN Override also applies to the VDN name forwarded using Information Forwarding. When a `consider` step is executed, the application number and available agent strategy assigned to the active VDN for the call is used.

VDN Return Destination (VRD)

Communication Manager clears the best resource data for a call when the call leaves vector processing and is not available if the call returns to vector processing.

BSR polling over IP without the B-channel interactions

- If you do not use a B-channel trunk facility, CMS records will poll attempts, but not actual trunk measurements.
- If you assign trunks, you can use the trunk group for H.323 IP voice calls, but you must install an IP media processor.

Call Prompting

Use Call Prompting for flexible call handling based on the information collected from a calling party. The information is in the form of dialed digits that originate from an internal or external touchtone telephone or from an internal rotary telephone that is on the same Communication Manager as the vector. Call Prompting facilitates temporary transfer of call management control to the caller.

When you enable Call Prompting and Call Vectoring, Communication Manager collects Caller Entered Digits (CED) and Customer Database Provided Digits (CDPD) from the network. Communication Manager receives Call Information Forwarding (CINFO) digits in the ISDN message of an incoming call when the AT&T Network Intelligent Call Processing (ICP) service is in use. Communication Manager then collects and forwards the digits to another Communication Manager using `interflow` commands.

With Voice Response Integration (VRI), you can use the **converse-on split** command to access a Voice Response Unit (VRU) script. The script returns the digits to Communication Manager. You can also use the digits for call management.

You can use Call Prompting in various applications to handle calls with more flexibility. Call Prompting uses specialized vector commands to process incoming calls based on information collected from the caller or from an ISDN-PRI message.

The following list provides a brief description of some Call Prompting applications:

- Automated Attendant: Allows the caller to enter the extension of the party the caller wants to reach, and routes the call to the desired extension.
- Data In/Voice Answer (DIVA) Capability: Allows the caller to hear an announcement based on the digits the caller enters, or to be directed to a hunt group or another system extension.
- Data Collection: Allows the caller to enter data that can be used by a host or adjunct to assist in call handling. The data, for example, can be the caller's account number.
- CINFO Routing: Allows call routing based on the digits the network supplies in an ISDN-PRI message.
- Message Collection: Allows the caller to leave a message or to wait in the queue for an agent.

Call Prompting command set

The following table show the commands that are used for Call Prompting.

Command category	Action	Command
Information collection	Collect information from the calling party, from the public network in an ISDN SETUP message, from a Voice Response Unit (VRU), or from CallVisor Adjunct Switch Application Interface (ASAI).	**collect digits**
Treatment	Play an announcement. Delay with audible feedback of silence, ringback, system music, or an alternate audio/music source.	**announcement** **wait-time**
Routing	Leave a message. Route the call to a number that is programmed in the vector. Route the call to digits that are supplied by the calling party.	**messaging split** **route-to number** **route-to digits**
Branching/ Programming	Go to a vector step. Go to another vector. Stop vector processing.	**goto step** **goto vector** **stop**

Touch-tone collection requirements

Ensure that Communication Manager has a collection resource to accept the touch-tone digits entered by a caller. The resource used for collecting and interpreting touch-tone digits is a unit of hardware called a Touch-Tone Receiver (TTR). The TTRs are provided on the call classifier and tone detector circuit packs, one of which is required for Call Prompting.

The number of TTRs that are required is configured according to the following sources:

- Customer input to the Avaya Account Team.
- Account team input to the configurator tool.

For existing systems that are adding a Call Prompting application, the appropriate number of TTRs can be based on two the following factors:

- Account team input to the configurator tool.
- Application review by the Avaya Design Center.

The process of collecting CINFO digits does not require TTRs.

Outside callers must have a touch-tone telephone to enter the digits that are requested by the **collect digits** command. For callers who are using rotary dialing, the Call Prompting time out takes effect, the **collect digits** command times out, and vector processing continues at the next step. As a precaution, always provide a default treatment such as a **route-to attendant** command, or a **queue-to split** command, in the vector script unless the script is created exclusively for users of touch-tone telephones.

Note:

You can administer any number of seconds from 2 to 10 for the **Call Prompting Interdigit Timeout** field on the Feature-Related System Parameters screen.

Call Prompting considerations

Consider the following points when working with Call Prompting:

- To enter the digits requested using a **collect digits** command, outside callers must have a touch-tone telephone. For such callers using rotary dialing, a 10 second inter-digit time out takes effect, and the **collect digits** command is omitted. As a precaution, a default treatment, for example, **route-to attendant** command, **queue-to split** command, must always be provided in the vector script unless the script is created exclusively for users of touch-tone telephones.
- If a caller does not enter the full number of digits specified in a **collect digits** step, an administered time out occurs. Thereafter, vector processing continues with subsequent vector steps and an attempt is made to process the call using the digits that have been collected. If the digits entered do not represent a valid destination, and if Automated Attendant is being implemented using a **route-to digits** command, the **route-to digits** command fails, and vector processing continues at the next step, which must be a default treatment.
- In case a **route-to attendant** command fails, provide a disconnect announcement.

- Before some vector steps, it is a good practice to insert a brief wait step so that vector processing can handle any possible delays that might cause the vector to fail and the call to not be routed. Follow this best practice when programming a vector:
  - Start all vectors with a “wait-time 0 seconds” step. Adding this step helps ensure you will have a touch-tone receiver (TTR) for vector processing.
  - When using an “adjunct routing” step, it is also advise to put a “wait-time 0 seconds” step before the “adjunct routing” step to provide a pause before routing the call to the adjunct system.
  - Any time digits are collected from a caller using a “collect digits” step, the step should be preceded by a “wait-time 0 seconds” step, hearing either ringback or music. Again, this ensures you will have a TTR to collect those digits.
- From time to time, all the touch-tone receivers can be in use. As a result, do not start the main vector with a **`collect digits`** command, since the caller on a DID or tie trunk in this case receives no audible feedback if the caller has to wait for a receiver to become available. Accordingly, include some treatment, for example, a **`wait-time 0 seconds hearing ringback`** step, before the initial **`collect digits`** step. The **`wait-time`** step is not necessary if the **`collect`** step is collecting ced or cdpd digits.

Call Prompting digit entry-collect digits command

Collect digits command

The **`collect digits`** command collects the touchtone digits entered by a Call Prompting user. The command allows the system to collect up to 24 digits from a touchtone phone. The first 16 digits are collected immediately, while the remaining digits are stored as dial-ahead digits.

Call Prompting offers some flexibility when entering digits. The caller can:

- Delete the incorrect digit string.
- Enter a variable-length digit string.
- Enter the dial-ahead digits.

Removing incorrect digit strings

About this task

You can include an announcement in the call treatment requesting the caller to enter digits. As an option, the announcement can instruct the caller to enter an asterisk (*) if the caller enters incorrect data.

When the caller enters an asterisk, the following happens:

Procedure

1. Digits collected for the current **`collect digits`** command are deleted.

Note:

Any dial-ahead digits entered, which do not exceed the maximum digit count of 24, are deleted. (Dial-ahead digits are explained later.)

2. Digit collection is restarted.
3. The announcement is not replayed.

Result

Once the caller enters an asterisk, the caller can re-enter digits for processing.

Variable-length digit strings

When administering the **`collect digits`** command, specify the maximum number of digits to be requested from the caller. Sometimes, you can prompt the caller to enter fewer digits than the maximum specified. In fact, the number of digits that the caller enters can vary for several variations of one **`collect digits`** command. Each such grouping of digits is called a variable-length digit string.

By using an end-of-dialing indicator in the form of the pound (#) sign, the Call Prompting permits variable-length digit strings does the following:

- Tells the system that the caller has finished entering digits.
- Causes the next vector step to be processed immediately.

When the caller is permitted to enter a variable-length digit string, the announcement part of the **`collect digits`** command specifies the largest possible number of digits that can be entered. Accordingly, each **`collect digits`** command must be administered to collect no more than the intended maximum number of digits. The caller can enter a pound sign part of a variable digit string entry either:

- At the end of each variable digit string that is entered. In this case, the pound sign must be included in the count of the number of maximum digits that can be entered.
- At the end of each such string that, not counting the pound sign, contains fewer characters than the maximum number of allowable digits. In this case, the pound sign must not be included in the count of the number of maximum digits that can be entered.

If the caller enters more digits than the maximum number specified, the additional digits are saved as dial-ahead digits for subsequent **`collect digits`** commands. If the vector or vectors chained to it do not contain another **`collect digits`** command, the extra digits are discarded.

If the caller enters fewer digits than the maximum number specified and does not complete the entry with the pound sign, a Call Prompting timeout occurs. The timeout ends the command and any digits collected before the timeout are available for subsequent vector processing. The Call Prompting timeout period is set to 10 seconds by default. But can be changed to a value between 2 to 10 seconds using the **Prompting Timeout** field on the Feature-Related System Parameters screen.

Caution:

Avaya recommends that you do not set the timeout to less than 4 seconds except in special cases. If the timeout is set to less than 4 seconds, the short timeout can cause the caller to miss entering the next digit in a sequence. The caller can miss entering the next digit in a sequence if the callers are unaware that the digits must be entered quickly. The setting of this timer is systemwide and affects digit entry for ALL collect digits steps in all vectors.

For more information, see “Collect digits command” in the *Programming Call Vectoring Features in Avaya Aura® Call Center Elite* document.

Using an application involving the entering of variable-length digit strings the user can dial either the number for the attendant or an extension to reach the wanted destination. If the maximum number of digits that can be entered is administered to be 3 and the user wishes to reach the attendant, the user must dial 0#. However, if the user chooses to dial a 3-digit extension, the user must dial, for example, 748 and not 748#. As the maximum number of digits that can be dialed in this case is three, dialing 748# causes the pound (#) sign to be saved as a dial-ahead digit. On the other hand, if the caller dials 748# and if the maximum number of digits that can be entered is four, the pound (#) sign is not saved as a dial-ahead digit as the sign is the last of four digits.

Dial-ahead digits

When digit collection for the current `collect digits` command is completed, vector processing continues at the next vector step. However, the switch continues to collect any digits that the caller subsequently dials until the TTR disconnects. For more information, see collecting digits on the switch in the *Programming Call Vectoring Features in Avaya Aura® Call Center Elite* document. The dialed-ahead digits are saved for processing by subsequent collect digits commands. Dial-ahead digits are explained fully in dial-ahead digits - collect digits command.

Functions and examples

Call Prompting uses some of the functions found in basic call vectoring. It also provides some additional functions that involve digit processing.

Treating digits as a destination

With Call Prompting, you can route calls according to the digits collected from the caller. Once the digits are collected by the `collect digits` command, the `route-to digits` command attempts to route the call to the destination that the digits represent. The command always routes the call to the destination indicated by the digits processed by the most recent `collect digits` command.

The digits can represent any of the following destinations:

- Internal (local) extension, for example split or hunt group, station, and announcement
- VDN extension
- Attendant
- Remote access extension

- External number such as a Trunk Access Code (TAC) or an Automatic Alternate Route/ Automatic Route Selection (AAR/ARS) FAC followed by a public network number, for example 7-digit Electronic Tandem Network (ETN), 10-digit DDD.

The following example shows how a call is routed by digits that are collected from a caller.

Using Call Prompting to route by collected digits

```
1. wait-time 0 seconds hearing ringback
2. collect 5 digits after announcement 300 [You have reached Redux Electric in
Glenrock. Please dial a 5-digit extension or wait for the attendant.]
3. route-to digits with coverage y
4. route-to number 0 with cov n if unconditionally
5. stop
```

In this vector, the caller is prompted to enter the destination extension of the party that the caller wants to reach (step 2). The extension in this vector can contain up to 5 digits. The vector collects the digits and routes to the destination by the **`route-to digits`** command in step 3.

If the **`route-to digits`** command fails because the caller fails to enter any digits, or because the extension number entered is invalid, the **`route-to number`** command in step 4 routes the call to the attendant, which is the default routing option. However, as long as the destination is a valid extension, the **`route-to digits`** command succeeds, coverage applies, and vector processing terminates. If the destination is busy, vector processing terminates because coverage call processing takes effect.

Note:

Occasionally, all of the system's TTRs can be in use. As a result, when you are collecting digits from a caller, do not start your main vector with a **`collect digits`** command, since the caller in this case receives no audible feedback if the caller has to wait for a TTR to become available. Include some treatment, for example wait-time 0 seconds hearing ringback, before the initial collect digits step.

In addition, if calls are to be transferred to this vector, add a wait-time step of sufficient length before the collect step to allow the transferring party time to complete the transfer.

Using digits to collect branching information

Call Prompting allows you to direct a call to another step or vector based on the digits that are entered by the caller. This branching is accomplished with a goto step. For example, in the following vector example, digits are used to route calls to different vectors based on an assigned customer number.

Using Call Prompting to branch by collected digits

```
1. wait-time 0 seconds hearing ringback
2. collect 5 digits after announcement 200 [
Please enter your customer number.
]
3. goto vector 8 if digits = 10+
4. goto vector 9 if digits = 11+
5. goto vector 10 if digits = 12+
6. route-to number 0 with cov n if unconditionally
7. stop
```

The wildcard + indicates that the two digits can be followed by zero or any number of additional digits. Callers with a number that begins with the digits 10 are routed to vector 8, callers with a number that begins with the digits 11 are routed to vector 9, and callers with a number that begins with the digits 12 are routed to vector 10.

Vector Routing Tables

You can test digits against entries in a Vector Routing Table. Vector Routing Tables contain lists of numbers used to test a **`goto...if digits`** command. Digits that are collected with the **`collect digits`** step can be tested to see if the digits are either in or not-in the specified table. Entries in the tables can include either the “+” or “?” wildcard.

- The “+” represents a group of digits and can only be used as the first or last character of the string.
- The “?” represents a single digit. Any number of “?” can be used at any position in the digit string.

Tables are entered on the Vector Routing Table screen. For information on Vector Routing Tables, see the *Administering Avaya Aura® Call Center Elite* document.

Use the following Call Vector example to test against the numbers provided in the Vector Routing Table.

Testing for digits in Vector Routing Table

```
1. wait-time 0 seconds hearing ringback
2. collect 7 digits after announcement 200 [“Please enter your account number.”]
3. goto vector 8 if digits in table 10
4. queue-to split 5 pri l
5. wait-time 10 seconds hearing ringback
6. announcement 2771
7. wait-time 10 seconds hearing music
8. goto step 6 if unconditionally
```

If the caller enters an account number listed in the Vector Routing Table, the call is routed to vector 8. If the caller enters an account number that matches the wildcard entry, for example 1345987, the call is routed to vector 8.

If the caller enters an account number that is not listed in the Vector Routing Table, or if the caller does not enter an account number, the call is queued to split 5.

Suppose that, instead of containing a list of premier accounts, the Vector Routing Table contains a list of accounts with a poor payment record. The following example shows a vector that only queues calls with account numbers that are not in the table. Calls in the table route to the collection department.

Testing for digits not in Vector Routing Table

```
 1. wait-time 0 seconds hearing ringback
 2. collect 7 digits after announcement 200 [“Please enter your account number.”]
 3. goto step 11 if digits = none
 4. goto step 6 if digits not-in table 10
 5. route-to number 83456 with cov y if unconditionally      [collections]
 6. queue-to split 5 pri l
 7. wait-time 10 seconds hearing ringback
 8. announcement 2771
 9. wait-time 10 seconds hearing music
10. goto step 8 if unconditionally
11. route-to number 0 with cov n if unconditionally
12. stop
```

If no digits are collected, the call is routed to the operator.

Note:

Entries in Vector Routing Tables can be tested against the telephone number of the caller Automatic Number Identification (ANI). For more information, see “ANI /II-digits routing and Caller Information Forwarding (CINFO)” in the *Programming Call Vectoring Features in Avaya Aura® Call Center Elite* document.

Using digits to select options

Call Prompting makes it possible to provide a menu of options that callers can use to satisfy their information needs. Callers select the desired option by entering the appropriate requested digit. Once the digit is entered, a conditional branch to the appropriate treatment is made. The treatment is usually provided by the **`route-to number`** command.

The following example shows how digits are used to select options.

Using Call Prompting to select options

```
1. wait-time 0 seconds hearing ringback
2. collect 1 digits after announcement 3531 [Thank you for calling Bug Out
Exterminators. If you wish to learn about the services we provide, please dial 1. If
you would like to set up an appointment for one of our representatives to visit your
home or place of business, please dial 2.]
3. route-to number 4101 with cov y if digit = 1
4. route-to number 4102 with cov y if digit = 2
5. route-to number 0 with cov n if unconditionally
6. disconnect after announcement none
```

In step 2 of this vector, the caller is asked to enter either 1 or 2, depending on the service used by the caller. If one of the digits is entered, an appropriate step (3 through 4) routes the call to the relevant extension, that is, either 4101 or 4102. If one of the digits is not entered, the call is routed to the attendant (step 5).

Displaying digits on an agent phone

If the agent has a two-line display phone and the display is in `normal` or `inspect` mode, the collected digits are automatically displayed on the second line. As a default, the digits remain on the line until the digits are overwritten, even after an agent releases the call. Administrators can decide when to clear the caller information on the agent phone display. For other types of phones, the agent must press the **callr-info** button to view the collected digits.

You can administer **callr-info** to reduce the agent time on the searching information that is related to the caller. For example, you can administer **callr-info** to collect information such as a customer account number, thereby eliminating the need for the agent to ask for the information.

Callr-info displays the caller information in the following format:

x = Info: 1234567890

where:

- x is a call appearance letter, for example a, b, and c.
- 1234567890 represents the digits that are collected from the caller.

The digits entered by the caller are collected by the most recent **`collect digits`** command. Any digits dialed ahead and not explicitly requested by the most recently executed **`collect digits`** command are not displayed.

If the agent presses **callr-info** when the call rings at the agent phone or when the station is active on a call appearance, the following events occur:

- The 10-second timer for display interval is set.
- The status lamp associated with the button is lit.
- The display is updated. Specifically, the incoming call identification, calling party ICI, is replaced with the collected digits in the format described earlier in this section. Only the digits collected for the last **`collect digits`** command are displayed.

If all the conditions to use the button, except for the collection of digits, are set and the agent presses the button, the status lamp associated with the button flashes denial.

More than one event can occur during a successful execution after the agent presses the button. The events include the following:

- The 10-second timer times out.
- The incoming call arrives at any call appearance.
- An active call changes status, for example another caller is added to the conference.

If any of the events occur, the following takes place:

- The status lamp associated with the button is turned off.
- The display is updated as previously described.

Note:

The agent can press **callr-info** again to view the digits again, provided the agent is active on the call or the call is still ringing. The agent can also flip between the collected digits and the ICI by alternately pressing **callr-info** and **normal**.

Clearing caller information from the station display

Administrators can decide when an agent's station display is cleared of caller information. Options include:

- Clearing the existing call information when the next call is received
- Clearing the existing call information when the call is released - whether the agent enters After Call Work (ACW) or not
- Clearing the existing call information when the agent leaves ACW mode or if the agent does not enter ACW, when the call is released

Passing digits to an adjunct

Call Prompting allows for the passing of information in the form of collected digits to an adjunct for further processing. Digits are passed to the adjunct by the ASAI Adjunct Routing capability.

An adjunct is any processor that is connected to a switch by the ASAI link. The adjunct makes a routing decision using the **`adjunct routing link`** command according to caller information and/or agent availability, and it returns the routing response to the switch. For example, the adjunct can indicate that the call be routed to a specific ACD agent. This is known as Direct Agent Calling (DAC).

A maximum of 16 Call Prompting digits from the last **`collect digits`** command can be passed to the adjunct by the **`adjunct routing link`** command.

The following example, shows how Call Prompting digits are passed to an adjunct.

Using Call Prompting to pass digits to an adjunct

```
1. wait-time 0 seconds hearing ringback
2. collect 10 digits after announcement 300 [Please enter your 10-digit account number]
3. adjunct routing link 15
4. wait-time 10 seconds hearing music
5. route-to number 52000 with cov y if unconditionally
6. stop
```

In step 2 of this vector, the caller is asked to enter a 10-digit account number. Once the account number is entered, the adjunct receives this information from the **`adjunct routing link`** command in step 3. This command then makes the appropriate routing decision if it is able to do so. If the command succeeds within the specified wait time, the command routes the call to the appropriate destination, and the call leaves vector processing. If the command fails, vector processing proceeds to the next step.

In addition to the adjunct routing capability, collected digits also can be passed by way of ASAI to an adjunct by prompting for the digits in one vector and then routing the call to a VDN that is monitored by an Event Notification (VDN) association. The collected digits (up to 16) are sent to the adjunct in a Call Offered to Domain Event Report. For detailed information, see *Communication Manager CallVisor ASAI Technical Reference*.

Note:

Adjunct Routing is fully discussed in Adjunct (ASAI) Routing.

Creating service observing vectors

Service observing vectors can be constructed to allow users to observe calls from a remote location or local station. When combined with Call Prompting, Service Observing vectors can route calls to a:

- Remote access service observing vector
- User-entered Feature Access Code (FAC) and extension
- Pre-programmed FAC and extension

Remote access Service Observing vector example

The following vector example connects a user to Remote Access. Once connected, the user can dial either a listen-only or listen/talk Service Observing FAC followed by the extension number to be observed. Although it is not required, Call Prompting increases security by providing passcode protection with remote service observing.

Remote access Service Observing vector

```
1. wait-time 0 secs hearing ringback
2. collect 5 digits after announcement 2300 [
Please enter your 5-digit      security
```

```
  code.
]
3. goto step 5 if digits = 12345 (security code)
4. disconnect after announcement 2000
5. route-to number 5000 with cov n if unconditionally
6. stop
```

User-entered FAC and extension example

The following vector example connects a user directly to the Service Observing FAC and extension based on the digits that are collected by Call Prompting.

Service Observing vector with user-entered FAC and extension

```
1. wait-time 0 secs hearing ringback
2. collect 5 digits after announcement 2300 [
Please enter your 5-digit      security
  code.
]
3. goto step 5 if digits = 12345 [security code]
4. disconnect after announcement 2000
5. wait-time 0 seconds hearing ringback
6. collect 6 digits after announcement 3245 [
Please enter the number 11 for
      listen-only observing or the number 12 for listen/talk observing      followed by
 the number of the extension you would like to observe.
]
7. route-to digits with coverage n
8. stop
```

Pre-programmed FAC and extension example

The following example shows a vector that connects a user to a pre-programmed FAC and extension using Call Prompting to allow the observer to select the extension that the observer wants to observe. In this example, the observer is Service Observing a VDN.

Service Observing vector with programmed FAC and extension

```
1. wait-time 0 secs hearing ringback
2. collect 5 digits after announcement 2300 [Please enter your 5-digit security code.]
3. goto step 5 if digits = 12345 [security code]
4. disconnect after announcement 2000
5. wait-time 0 seconds hearing ringback
6. collect 1 digits after announcement 2310 [Enter 1 to observe sales, 2 to observe
billing.]
7. route-to number 113001 with cov n if digit = 1 [11 = listen-only observe, 3001 =
Sales VDN]
8. route-to number 113002 with cov n if digit = 2 [11 = listen-only observe, 3002 =
Billing VDN]
9. goto step 6 if unconditionally
```

Dial-ahead digits-collect digits command

Dial-ahead digits

With Dial-ahead digits, callers can skip announcements. Only subsequent **`collect digits`** commands can use the digits. Other vector commands that operate on digits, for example, **`route-to digits`** and **`goto...if digits`**, do not use the digits until Communication Manager collects the digits.

Communication Manager does not forward the digits with interflowed calls and does not display the digits to an agent who presses **callr-info** until Communication Manager completes the **`collect digits`** command operation.

Collection of dial-ahead digits continues until one of the following occurs:

- Vector processing stops or is terminated.
- The sum of the digits collected for the current **`collect digits`** command plus the dial-ahead digits exceeds the storage limit of 24. Communication Manager discards additional digits until subsequent **`collect digits`** commands create more storage.

  Note:

  Asterisk (*) and pound sign (#) digits are part of the 24 digit limit, as do any dial-ahead digits that callers enter after the asterisk (*) or pound sign (#) digit.
- The TTR required by the caller to collect digits is disconnected, which can happen when one of the following conditions is true:
  - A successful or unsuccessful **`route-to number`** step is encountered during vector processing, except where the number routed to is a VDN extension.
  - A successful or unsuccessful **`route-to digits`** step is encountered during vector processing, except where the number routed to is a VDN extension.
  - A successful or unsuccessful **`adjunct routing link`** step is encountered during vector processing.
  - A successful or unsuccessful **`converse-on`** step is encountered during vector processing.
  - A Call Prompting time out occurs, during which time the caller does not dial additional digits, asterisks (*) sign, or pound sign (#).
  - Vector processing stops or is terminated.
  - A successful or unsuccessful collect ced step or cdpd step is encountered.

  When the TTR is disconnected due to a **`route-to number`**, **`route-to digits`**, **`converse-on`**, **`adjunct routing link`** , or **`collect ced/cdpd`** step, Communication Manager discards all dial-ahead digits. Therefore, following a failed **`route-to`**, **`converse`**, or **`adjunct routing link`** step, a subsequent **`collect digits`** step always requires callers to enter digits.

Dial-ahead digit vector examples

The vectors in the following examples illustrate a situation where a caller can enter dial-ahead digits. In this case, the caller must have a touch-tone telephone. Program an alternative handling sequence in case the caller has a rotary telephone or the caller does not dial a touch-tone digit before the time out period.

Step 2 of vector 30 offers the caller two options, each of which provides different information. The caller is prompted to enter either 1 or 2, depending on the information the caller wants to hear. Once the caller enters a digit, the digit is collected by the **`collect digits`** command. Thereafter, an attempt is made by the **`route-to number`** command to route the call to the appropriate vector (step 3 or 4). If the caller enters a digit other than 1 or 2, the appropriate announcement is provided (step 5), and the digit entry cycle is repeated (step 6).

If the caller enters 1, vector 31 is accessed.

Using dial-ahead digits to bypass announcements, example 1

```
VDN (extension=1030   name=''Coastal''   vector=30)
Vector 30:
1. wait-time 0 seconds hearing ringback
2. collect 1 digits after announcement 3000 [Thank you for calling Coastal League
Baseball Hotline. You must have a touch-tone telephone to use this service. If you wish
to hear the scores of yesterday's games, please press 1. If you wish to hear today's
schedule of games, please press 2.]
3. route-to number 1031 with cov y if digit = 1
4. route to number 1032 with cov y if digit = 2
5. announcement 301 [Entry not understood. Please try again.]
6. goto step 2 if unconditionally
```

In step 1 of vector 31, the caller is offered three options that supplement the original option provided in vector 30. The caller is prompted to enter either 3, 4, or 5, depending on what information the caller wants to hear. If the caller enters an incorrect digit, the customary digit correction routine is implemented (steps 5 and 6). Once an appropriate digit is entered, the call is routed, in this example by a **`goto step`** command (step 2, 3, or 4), to the appropriate announcement (step 7 or step 9).

In step 10 of vector 31, the caller is prompted with the choice of returning to the main menu provided in vector 30 or of terminating the call. If the caller selects the former option (by entering 9), the call is routed to vector 30 and the entire process is repeated.

Using dial-ahead digits to bypass announcements, example 2

```
VDN (extension=1031  name=''Scores''   vector=31)
Vector 31:
 1. collect 1 digits after announcement 4000 [If you wish to hear scores of games in
both divisions, please press 3. If you wish to hear scores for Northern Division games
only, please press 4. If you wish to hear scores for Southern Division games only,
please press 5.]
 2. goto step 7 if digits = 3
 3. goto step 7 if digits = 4
 4. goto step 9 if digits = 5
 5. announcement 301 [Entry not understood. Please try again.]
 6. goto step 1 if unconditionally
 7. announcement 4002 [Northern Division scores]
 8. goto step 10 if digits = 4
 9. announcement 4003 [Southern Division scores]
10. collect 1 digits after announcement 4004 [If you wish to return to the main menu,
please press 9. Otherwise, press 0.]
11. route-to number 1030 with cov n if digit = 9
12. goto step 15 if digit = 0
13. announcement 301 [Entry not understood. Please try again.]
14. goto step 10 if unconditionally
15. disconnect after announcement none
```

Vector 32 is similar in design to vector 31. The major difference is the information provided and the requested digit entries.

In this example, the caller has to go through two sets of options to get the desired information. Each option set is introduced by an announcement. However, because of the dial-ahead digit capability, the caller can bypass the announcements if the caller wishes to. The caller can enter 1 and 5 within a matter of seconds to hear yesterday's Southern Division scores.

The caller can enter digits while the call is being queued for an announcement or while the announcement is playing. If digits are entered during an announcement, the announcement is disconnected. If digits are entered while a call is queued for an announcement, the call is removed from the announcement queue.

Dial-ahead digits, example 2

```
VDN (extension=1032   name=Schedule   vector=32)
Vector 32
 1. collect 1 digits after announcement 5000 [If you wish to hear today's schedule of
games in both divisions, press 6.
    To hear today's schedule of games in the Northern Division only, press 7. To hear
today's schedule of games in the
    Southern Division only, press 8.]
 2. goto step 7 if digits = 6
 3. goto step 7 if digits = 7
 4. goto step 9 if digits = 8
 5. announcement 301 [Entry not understood. Please try again.]
 6. goto step 1 if unconditionally
 7. announcement 5002 [Northern Division schedule]
 8. goto step 10 if digits = 7
 9. announcement 5003 [Southern Division schedule]
10. collect 1 digits after announcement 4004 [To return to the main menu, press 9.
Otherwise, press 0.]
11. route-to number 1030 with cov n if digit = 9
12. goto step 15 if digits = 0
13. announcement 301 [Entry not understood. Please try again.]
14.goto step 10 if unconditionally
15.disconnect after announcement none
```

ASAI-requested digit collection

With ASAI-requested Digit Collection, an adjunct can request that a DTMF tone detector be connected for the purpose of detecting user-entered digits. The collected digits are passed to ASAI monitoring or controlling adjuncts for action. Communication Manager handles the digits as if the digits were dial-ahead digits. The caller can request for Sequence Dialing after the call has been routed to the final destination and has resulted in an unanswered call, that is busy, no answer.

The digits are not necessarily collected while the call is in vector processing. The digits can be sent to an ASAI adjunct, used by Call Prompting features, or both.

You must enable **ASAI Adjunct Routing** and **Call Prompting** for the feature to work.

ASAI-provided dial-ahead digits - collect digits command

The ASAI-provided digits feature allows an adjunct to include digits in a Route Select capability. The digits are treated as dial-ahead digits for the call. Dial-ahead digits are stored in a dial-ahead digit buffer and can be collected one at a time or in groups using the **`collect digits`** command. Although the adjunct can send more than 24 digits in a Route Select, only the first 24 digits (or 24-`x`, where`x` is the number of digits collected by vector processing prior to executing the **`adjunct routing link`** vector command) are retained as dial-ahead digits. An application can use this capability to specify the digits that Communication Manager must pass to the VRU as part of the **`converse-on`** vector step.

Note:

The maximum number of dial-ahead digits that can be stored in the buffer is dependent on the number of digits that were already collected for the call by a previous **`collect digits`** vector command. If `x` digits were collected by vector processing prior to executing an **`adjunct routing link`** vector command, the `x` digits collected reduces the maximum number of digits that can be stored as dial-ahead digits as a result of a Route Select. The rest are discarded.

Call Prompting interactions

The following interactions apply specifically to Call Prompting. For general Call Vectoring interactions that affect Call Prompting applications, see "Call Vectoring".

Authorization Codes

If authorization codes are enabled, and a **`route-to`** command in a prompting vector accesses AAR or ARS, if the VDN's FRL does not have the permission to use the chosen routing preference, then the system does not prompt for an authorization code and the **`route-to`** command fails.

CallVisor ASAI

ASAI-provided digits can be collected by the Call Vectoring feature using the **`collect`** vector command as dial-ahead digits. CINFO is passed to CallVisor ASAI.

Hold

With the exception of CINFO, if a call is put on hold during the processing of a **`collect`** command, the command restarts, beginning with the announcement prompt, when the call is taken off hold. All dialed-ahead digits are lost. Similarly, if a call to a vector is put on hold, vector processing is suspended when a **`collect`** command is encountered. When the call becomes active, the **`collect`** command resumes.

Inbound Call Management (ICM)

You can use Call Prompting to collect information that can be used by an adjunct to handle a call.

Transfer

If a call to a VDN is transferred during a **`collect`** command, the **`collect`** command restarts when the transfer is complete, and all dialed-ahead digits are lost. Similarly, if a call to a vector is transferred, vector processing is suspended when a **`collect`** command is encountered. When the transfer is complete, the **`collect`** command resumes. This is not true when a **`collect`** command collects CINFO digits. In this case vector processing is not suspended. Attendant extended calls do suspend vector processing in the same way as transferred calls.

Holiday Vectoring

Holiday Vectoring simplifies vector writing for holidays. With Holiday Vectoring, you can route calls or provide special handling for date-related calls on a regular basis. You can administer as many as 999 different holiday tables and use the tables to make vectoring decisions. Each table can have up to 15 dates or date ranges. You can administer the Holiday Vectoring capabilities in advance to ensure seamless call routing during holidays when staffing reduces, or when call centers close.

When vector processing encounters a `goto` xxx if holiday in table # step, vector processing determines if the current date and time qualifies as a holiday in the specified table. The information is then used to decide whether the goto condition is true or false and in turn to decide whether to go to the given step or not. The date and time match is done at the time that the call is in vector processing. The date and time match is similar to the match done in the Time-of-Day (TOD) routing, which means that the system date and time is checked on the Processor Port Network (PPN), rather than the local port network time on the Expansion Port Network (EPN).

Holiday Vectoring is not limited to holiday use, but can also be applied to any date-related special processing. For example, vectors can be modified or created to perform special processing during a two-week television promotion or a semi-annual sale.

The Holiday Vectoring feature benefits administrators who have to administer as many as 30 bank holidays throughout the year. Holiday Vectoring streamlines vectoring tasks and ensures seamless operation during holidays or special events.

Without Holiday Vectoring, call center administrators have to write special vectors for each holiday or other special date related circumstance and ensure that the vectors are administered at appropriate times. In some cases, administrators are required to go to work on holidays just to administer vectors.

Administering Holiday Vectoring

About this task

To enable the **Holiday Vectoring** field you must first enable either **Vectoring (Basic)** or **Attendant Vectoring**.

You can use up to 999 different holiday tables if you enable the **Vectoring (3.0 Enhanced)** field. Otherwise, you can use up to 10 holiday tables.

On the Call Center pages of the System-Parameters Customer-Options screen, set the **Vectoring (Holidays)** field to `y`.

Holiday Vectoring considerations

- Administration of holiday tables is supported only on Communication Manager and cannot be changed using adjunct vectoring tools.
- Holiday vectoring is only available when you enable **Vectoring (Basic)** or **Attendant Vectoring**.
- There is no procedure to validate consistency among the 15 holidays in any table. If the same holiday is entered twice, the system stops checking after the first found entry.
- Date range overlapping is common for holidays that are in ranges of dates. When a call is in vector processing, holidays are checked from top to bottom on the table and the check stops

if a match is found. Even though there can be multiple matching entries, the check stops at the first match.

- There is a validation that the day of the month that is entered is valid with the given month. Specifically, if the month is April, June, September, or November, then the date must be a number between 1 and 30. If the month is January, March, May, July, August, October, or December, then the date can be a number between 1 and 31. If the month is February, a the date can be any number between 1 and 29.

  Note:

  The year is not checked in holiday vector processing. This allows the same holidays to be used year-to-year when the holiday is on a fixed date. For holidays where the date changes from year-to-year, you must re-administer the holiday tables.

- When disabling the Holiday Vectoring feature, that is, changing the value of the **Vectoring (Holidays)** field from `y` to `n` on the System-Parameters Customer-Options screen, the vectors are checked for any **`goto if holiday`** steps. If any such steps are found, an error message is displayed and the change is not allowed. You must remove the vector steps first before you disable the Holiday Vectoring.
- You can use the **Use VDN Time Zone For Holiday Vectoring** field on the Vector Directory Number (VDN) screen that is active for the call so that vector processing uses the VDN time zone specified when applying holiday vectoring if the field is set to `y`, otherwise vector processing uses the system time.

Holiday Vectoring command set

Holiday Vectoring enables a set of commands that are used to create call vectors, which route calls on holidays or days when special processing is required. The following table shows the commands that are available for use in Holiday Vectoring.

Command category	Action taken	Command
Branching/programming	Go to a vector step	**`goto step`**
	Go to a vector	**`goto vector`**

Goto step command for Holiday Vectoring

Syntax and valid entries

Command name	Parameter	Variables	Condition	Variables	Description
`Goto`	step	<step #> 1-99	if holiday in table	<table #> 1-999, A-Z, AA-ZZ, V1-V9	This command directs the call to a specific vector step if the conditions of the call match a holiday that is in the specified holiday table.

Table continues…

Command name	Parameter	Variables	Condition	Variables	Description
			if holiday not in table		This command directs the call to a specific vector step if the conditions of the call do not match any of the holidays that are in the specified holiday table.

Goto vector command for holiday vectoring

Syntax and valid entries

Command name	Parameter	Variables	Condition	Variables	Description
`Goto`	vector	<vector #> 1-8000 @ step 1–99	if holiday in table	<table #> 1-999, A-Z, AA-ZZ, V1-V9	This command directs the call to a specific vector if the conditions of the call match a holiday that is in the specified holiday table.
			if holiday not in table		This command directs the call to a specific vector if the conditions of the call do not match any of the holidays that are in the specified holiday table.

Holiday table command syntax

Syntax 1

```
change holiday-table x
```

This command allows you to change the entries in a holiday table. To create a new holiday table, you must use the **change** command with a table number that is not in use. For example, change holiday-table 9, where table 9 has not been used to define holidays.

Syntax 2

```
display holiday-table x
```

This command allows you to view the entries in the specified holiday table.

Syntax 3

```
list holiday-table
```

This command lists all of the holiday tables in use.

Syntax 4

`list usage holiday-table x`

This command lists all vector steps that refer to the selected holiday table.

Using the holiday table commands

About this task

After enabling **Vectoring (Holidays)** on the Customer Options screen, enter:

`change holiday-table 1`

On the Holiday Table screen, which is shown in the following example, enter the holiday information.

Setting up a Holiday Table

```
change holiday-table 1                                     page 1 of 1
                               HOLIDAY TABLE
    Number: 1              Name: Bank Holidays
          START                     END
    Month Day Hour Min     Month Day Hour Min     Description
    12     24  00   00     12     31  23   59     Christmas
    01     01  00   00     01     02  06   00     New Year's Day
```

Note:

When using a range of dates, the end date must be greater than the start date. Ranges must be within one calendar year. There are two entries in the example, one for each calendar year.

The Holiday Table screen can be used to enter individual holidays or holiday ranges. The following rules apply for entering dates on this screen:

Procedure

1. If a day is entered, the corresponding month must be entered.
2. If a month is entered, the corresponding day must be entered.
3. If an hour is entered, the corresponding minute must be entered.
4. If a minute is entered, the corresponding hour must be entered.
5. If an hour and minute is entered, the corresponding month and day must be entered.
6. If a month and day is entered, the corresponding hour and minute is not required.
7. If an end month and day is entered, the corresponding start month and day must be entered.
8. If a start month and day is entered, the corresponding end month and day is not required.
9. To enter an individual holiday, enter a start month and day, but do not enter an end month and day.
10. To enter a holiday range, enter both a start month and day and an end month and day.

11. The start month, day, hour, and minute must be less than or equal to the end month, day, hour, minute.
12. The description field is alphanumeric.

Result

After creating a holiday table, use the **`display holiday-table`** command to view the entries. To list all of the holiday tables, use the **`list holiday-table`** command, as shown in the following example.

Listing Holiday Tables

```
list holiday-table
                                HOLIDAY TABLES

                          Table Number      Name
                              01            Business Holidays
                              02            Annual Promotion Dates
                              03            Summer Special
                              04
                              05
                              06
                              07
                              08
                              09
                              10
```

Changing vector processing for holidays

About this task

You must add or change vector processing for the administered holiday tables.

On the command line, enter `change vector` x (where x is the vector number). The Call Vector screen appears with editable fields. Customers can enter a new goto conditional for the holidays.

When you set the **Holiday Vectoring** field to `y`, a field on the Call Vector screen identifies if the vector on which you are currently working is a holiday vectoring vector, as shown in the following example.

```
change vector x                                                  page 1 of 3

                                      CALL VECTOR


    Number: xxx      Name: ___________________________
Multimedia? n      Attendant Vectoring? n    Meet-me Conf? n           Lock? y
     Basic? y   EAS? n   G3V4 Enhanced? y   ANI/II-Digits? y   ASAI Routing? y
 Prompting? y   LAI? y  G3V4 Adv Route? y   CINFO? y   BSR? y   Holidays? y
 Variables? y               3.0 Enhanced? y

01 ____________
02 ____________
03 ____________
04 ____________
05 ____________
06 ____________
07 ____________
08 ____________
09 ____________
```

```
10 ____________
11 ____________
```

The **Vectoring (Holiday)** field is a display-only field and appears only when you set the **Holiday Vectoring** field on the Customer Options screen to `y`. If either **Vectoring (Basic)** or **Attendant Vectoring** are set to `y`, the **Vectoring (Holiday)** field can be set to `y`.

The following examples use the **`goto`** command to route calls during holidays.

Example 1

```
 change vector 1                                              Page 1 of 3
                                  CALL VECTOR

    Number: 1        Name: In Germany
Multimedia? n      Attendant Vectoring? n    Meet-me Conf? n           Lock? y
     Basic? y   EAS? n   G3V4 Enhanced? n   ANI/II-Digits? n   ASAI Routing? n
 Prompting? y   LAI? n  G3V4 Adv Route? n   CINFO? n   BSR? n   Holidays? y

01 goto         vector 2  if holiday         in    table 1
02 route-to     number 123456789             with cov n if unconditionally
03
04
05
06
07
08
09
10
11
```

Example 2

```
change vector 3                                              Page 1 of 3
                                  CALL VECTOR

    Number: 3        Name: In Ireland
Multimedia? n      Attendant Vectoring? n    Meet-me Conf? n           Lock? y
     Basic? y   EAS? n   G3V4 Enhanced? n   ANI/II-Digits? n   ASAI Routing? n
 Prompting? y   LAI? n  G3V4 Adv Route? n   CINFO? n   BSR? n   Holidays? y

01 goto         step   2  if holiday         in    table 2
02 route-to     number 45678          with cov n if unconditionally
03 stop
04 announcement 2721
05
06
07
08
09
10
11
```

After you have assigned holiday tables to several vectors, you can use the **`list usage holiday-table`** command to view which vectors and vector steps are using the selected holiday table.

```
list usage holiday-table x
                              LIST USAGE REPORT

     Used By
     Vector                Vector Number 1                      Step 1
     Vector                Vector Number 3                      Step 1
```

Look-Ahead Interflow

Look-Ahead Interflow (LAI) enhances Call Vectoring for call centers with multiple ACD locations. With LAI, call centers can improve their call-handling capability and agent productivity by intelligently routing calls among call centers to achieve an improved ACD load balance. This service is provided by ISDN D-channel messaging over QSIG or non-QSIG private networks, virtual private networks, or public networks. LAI is also supported on SIP trunks. The receiving Communication Manager can accept or deny interflowed calls sent by the sending Communication Manager.

LAI has the following basic attributes:

- Produces First in First Out (FIFO) or near-FIFO call processing
- Includes enhanced information forwarding, that is, codeset 0 user information transport

LAI requirements

The following items are criteria for basic LAI call control operation over a VPN or a public switched network:

- The sending and receiving call center locations must have ISDN PRI or BRI trunk facilities.

  Note:

  ATM trunking and IP trunking can be set up to emulate ISDN PRI, but first verify that the release of Communication Manager that you have still supports the use of ATM trunking.
- The switch must support the ISDN country protocol.
- LAI has been tested with several major carriers. To find out if the capabilities work with your carrier, check with your account team for the latest information. If testing is not done to verify operation over the public networks involved with the preferred specific configuration, use private ISDN trunking between the nodes until successful testing is complete.
- The ISDN SETUP and DISCONNECT messages are transported between sending and receiving locations, for example, Signaling System 7 (SS7) or equivalent public network connectivity.
- A receiving-end generated DISCONNECT message must transmit back to the sending switch call center without changing the cause value.

  Conversion of the DISCONNECT message to a progress message, with a **Progress Indicator Description** set to `1` and a cause value other than 127 included, is a valid reject message and compatible with LAI.
- Progress messages that are generated towards the sending end by intervening network switches must have the **Progress Indicator Description** set to `8` so that the switch does not treat the call accepted or rejected.
- ISDN codeset 0 user information transport supports LAI information forwarding. As an alternative, LAI can use dedicated VDNs at the receiving location to provide an equivalent display of the forwarding application identity and set trunk group options to not send either the codeset 6/7 LAI IE or codeset 0 information transport.

Note:

BSR cannot use the LAI alternatives. BSR must use ISDN codeset 0 user information transport.

LAI considerations

Consideration	Description
Carrier compatibility	LAI works with several major carriers. To find out if the LAI capabilities work with your carrier, check with your account team for current information. If testing has not been done to verify operation over the public networks involved with the preferred specific configuration, use private ISDN trunking between the nodes until successful testing is complete. ISDN routing with LAI enabled: All calls routed over ISDN facilities by a **`route-to number`** with cov `n` or **`route-to digits`** with cov `n` vector command on a communication server where LAI is enabled are treated as LAI call attempts. A vector can route a call over an ISDN facility to a destination that is not a VDN. The sending communication server processes this call as an LAI call even if the call is not an LAI call. ISDN processing at the receiving communication server causes the call to always be accepted. However, the DNIS and any other information in the LAI information forwarded with the call are ignored. Interim call handling before LAI is accepted by receiving communication server: Until the LAI attempt is accepted by the receiving communication server, the caller continues to hear any feedback applied by the sending communication server vector and remains in any split or skill queue.
Call handling with Route-to number or Route-to digits handling with coverage	**`Route-to number`** with cov `y` or **`route-to digits`** with cov `y` commands does not result in an LAI call attempt. The commands always complete the call. Do not use the command if the vector at the receiving communication server can deny the call, since the caller in this case hears a busy signal. Use this command with coverage `y` only when you want unconditional interflow with LAI active and the terminating communication server is set up accordingly.
Continuity during call transfer between communication servers	Audible feedback can be provided to the caller before interflow is attempted. Therefore, another audible feedback from the receiving communication server can confuse the caller. For example, a caller hearing ringback on the sending communication server can be confused if music is applied suddenly when the call interflows to the receiving communication server.

Table continues…

Consideration	Description
Backward compatibility of LAI applications	For backward compatibility of LAI applications between Avaya communication servers, leave the **Send Codeset 6/7 LAI IE** field on the Trunk Group screen to the default `y`. Existing LAI applications continue to operate as before, even after you upgrade. You can use enhanced LAI available in the communication server without any network or trunk administration changes, by adding the “interflow-qpos” conditional to original LAI vectors.
AAR/ARS	ISDN facilities used to provide LAI to a VDN on another communication server in a private network can use the AAR feature if private facilities are to be used for call routing.
Agent Telephone Display	If collected digits are forwarded with an interflowed call, the forwarded digits are displayed to the answering agent on the telephone display.
Attendant Control of Trunk Group Access	Calls do not route over a trunk with Attendant Control of Trunk Group Access set.
Authorization Codes	Authorization codes are not required for interflow routing. Assign a high enough FRL to the VDN so that the route desired for routing interflow calls can be used without requiring an authorization code entry. If a route choice is encountered that requires a higher FRL, the interflow is an invalid destination, rejected for LAI or not available for standard interflow, without the application of recall dial tone.
BCMS	Neither does BCMS log LAI attempts nor does BCMS report accumulated in-VDN time.

Table continues…

Consideration	Description
Call Detail Recording - Sending Server	No ineffective call attempt or outgoing call CDR records are generated for vector **`route-to`** commands that are unsuccessful including denied LAI attempts. If a local call to a VDN generates an LAI call attempt that is accepted and answer supervision is returned from the receiving communication server, one outgoing call CDR record is generated with the originating extension as the calling number. If an incoming call to a VDN generates an LAI call attempt that is accepted, and no answer supervision is returned from the receiving communication server, one incoming CDR record is generated. The VDN is the called number and the duration is from the time answer supervision was provided to the incoming trunk. If an incoming call to a VDN generates an LAI call attempt that is accepted and answer supervision is returned from the receiving communication server, two incoming CDR records are generated: • An incoming record with the VDN as the called number and the duration as the time since answer supervision was provided to the incoming trunk. This is generated if the call is initially answered in the sending communication server before interflow takes place. • An outgoing record containing the incoming trunk information as the calling number and the dialed digits and the outgoing trunk information as the called number.
Call Detail Recording - Receiving Server	On the receiving communication server, an incoming LAI call is treated like any other incoming vector call. If answer supervision is returned by the vector and the call does not terminate to another destination, the VDN extension is recorded as the called number in the CDR record. If the call terminates to a hunt group, the VDN, hunt group, or agent extension is recorded as the called number. If the **Record VDN in Record** field on the Feature-Related System Parameters screen is `y`, the VDN extension overrides the **Call to Hunt Group - Record** field entry for vector calls.
Call Prompting	Digits collected at the sending Communication Manager, no matter how the digits are collected (caller-entered, ASAI provided, CINFO provided) are forwarded with interflowed calls and available at the remote Communication Manager using Information Forwarding. **Note:** Dial-ahead digits are not forwarded with the call. There is a maximum of 16 forwarded digits.
Centralized Attendant Service	A centralized attendant can be an LAI destination.

Table continues…

Consideration	Description
VDN Name Display	The VDN name, part of the LAI information forwarded with calls, can be up to 27 characters long. Any characters over this limit is dropped. On Communication Manager, a VDN name can be administered with as many as 27 characters.
Distributed Networking - Manufacturers Specific Information (MSI)	LAI, whether enhanced or no, does not function with systems from other vendors unless that vendor develops a corresponding capability that works with the Avaya communication server.
Facilities Restriction Level and Traveling Class Marks	The FRL for interflow over ARS/AAR route choices is assigned to the original VDN used for the incoming call.
Incoming Call Management	You can use the adjunct routing capabilities of vectoring at the sending communication server to determine if a call must be interflowed. You can use the adjunct routing at the receiving communication server to tandem the call to a far-end communication server. If the call terminates to a tandem trunk, two CDR records are generated: • An incoming record with the VDN as the called number and the duration as the time since answer supervision was provided to the incoming trunk. • An outgoing record containing the incoming trunk information as the calling number and the dialed digits and the outgoing trunk information as the called number.
Network Access	LAI operates over public, private, or virtual private networks. ISDN-BRI and -PRI networks that meet minimum network requirements. The sending of an LAI codeset 6/7 information element is counted toward Message Associated User-to-User Information (MA-UUI) counts.
Path replacement for QSIG/DCS ISDN calls	For calls that are waiting in queue or in vector processing, even if the call is not connected to an answering user, path replacement using QSIG can be attempted to find a more optimal path for this call. This results in more efficient use of the trunk facilities. The QSIG ISDN or DCS ISDN trunk path replacement operation can be triggered for ACD calls by the LAI **`route-to number`** vector step, BSR **`queue-to best`** vector step, and the adjunct routing vector steps.
QSIG	LAI and information forwarding function over QSIG trunk facilities if the remote locations are Avaya communication servers. You can get LAI call control functionality with other vendors if an Avaya communication server is the starting point.
RONA	Calls redirected to a VDN by RONA can subsequently be processed and routed by LAI applications.
Service Observing	You can observe a call in LAI processing using VDN observing throughout the life of the call as long as the call is still connected through the local communication server. All current restrictions on Service Observing still apply. Incoming calls can be service observed at the remote communication server.

Table continues…

Consideration	Description
Trunk-to-Trunk Transfer	Interflowed calls can be transferred by a receiving communication server to another trunk connection.
VDN Override	The name of the active VDN for a call is displayed at the remote answering agent.

Two switch configuration

LAI is enabled through the use of call vectors and their associated commands. For a two-switch configuration, these vectors are included in both the sending switch, which processes vector outflow, and the receiving switch, which processes vector inflow.

LAI command set

The following table lists the call-acceptance vector commands that are used in LAI.

Command	Qualification
`announcement`	Announcement available Queued for announcement Retrying announcement
`check split`	Call terminates to agent Call queued to split
`collect digits`	Always, except for ced and cdpd digits, which are neutral
`converse-on split`	VRU answers the call Call queued to converse split
`disconnect`	With announcement and announcement available With announcement and queued for announcement With announcement and retrying announcement
`messaging split`	Command successful Call queued
`queue-to split`	Call terminates to agent Call queued to split
`route-to`	Terminates to valid local destination Successfully seizes a non-PRI trunk Results in a LAI call attempt, and the call is accepted by the far-end switch
`wait-time`	Always, except **`wait-time hearing i-silent`**, which is neutral

If the receiving Communication Manager is unable to accept the LAI call, one of the vector steps executes call denial.

Note:

Use the **`busy`** command instead of the **`disconnect`** command to allow for compatibility with similar network services such as Alternate Destination Redirection (ADR).

Vector command (for call denial)	Qualification
`busy`	Always
`disconnect`	Without announcement With announcement but announcement unavailable
`reply-best`	Always. Used with BSR

The vector commands in the following table are treated as neutral because the commands do not generate call acceptance or denial messages.

Neutral vector command	Qualification
`adjunct routing link`	Always
`announcement`	Announcement unavailable
`check split`	Call neither terminates nor queues
`collect ced/cdpd digits`	Always
`consider`	Always. Used with BSR
`converse-on split`	Call neither terminates nor queues
`goto step`	Always
`goto vector`	Always
`messaging split`	Command failure
`queue-to split`	Call neither terminates nor queues
`route-to`	Unsuccessful termination Trunk not seized LAI call denied by the far-end switch
`stop`	Always
`wait-time hearing i-silent`	Always **Note:** This command is used following an **`adjunct routing link`** command in applications where the adjunct decides whether to accept or reject the Look-Ahead calls.

How traditional LAI works

Use the traditional LAI feature when the preferred call flow performs LAI attempts before queuing the call.

LAI commands

LAI uses the commands included within the basic Call Vectoring and Call Prompting features:

- **`route-to number with coverage n`** or **`route-to digits with coverage n`** command on Communication Manager that has LAI enabled and that successfully seizes an ISDN trunk automatically results in a normal LAI call attempt being placed. The call attempt can be rejected or accepted by the remote end.
- **`route-to number with coverage y`** or **`route-to digits with coverage y`** command does not result in an LAI call attempt. The command always completes the call. Do not use the command when the vector at the receiving location ends up denying the call since the caller will hear a busy signal, or the call will disconnect. Use the command with coverage set to `y` only for those cases when an unconditional interflow is wanted (with LAI active) and the terminating Communication Manager is set up to handle this type of call.

When a LAI call attempt is made, Call Vectoring at the sending location checks a potential receiving location to determine whether to hold or send the call. While this is done, the call remains in queue at the sending location. As such, the call can still be connected to the sending-location agent if one becomes available before the receiving location accepts the call.

Call Vectoring at the receiving location determines whether to accept the call from the sending location or to instruct the sending location to keep the call. In the latter case, the sending location can keep the call, check other locations, or provide another call treatment. Conditions for sending, refusing, or receiving a LAI call attempt can include a combination of any of the following:

- Expected wait time for a split.
- Number of staffed or available agents for a skill.
- Number of calls in queue.
- Average speed of answer or the number of calls active in a VDN.
- Time of day and day of week.
- Any other legitimate conditional.

If the call is accepted by the receiving Communication Manager, the call is removed from any queues at the sending Communication Manager and call control is passed to the receiving Communication Manager. If the call is denied by the receiving Communication Manager, vector processing continues at the next step at the sending Communication Manager. Until the call is accepted by either Communication Manager, the caller continues to hear any tones applied by the sending Communication Manager. If the call is denied, the call vector can apply alternate treatment, such as placing another LAI call to an alternate backup Communication Manager.

Note:

The LAI operation is completely transparent to the caller. While a LAI call attempt is being made, the caller continues to hear any audible feedback that is provided by the sending Communication Manager vector. The caller also maintains position in any split queues until the call is accepted at the receiving Communication Manager.

LAI passes Call Prompting digits collected in the sending Communication Manager to the receiving Communication Manager by codeset 0 user information transport.

Example of traditional LAI

The vectors in the sending Communication Manager use the **goto** command to determine whether the call must be sent to the receiving Communication Manager. Recall that the **goto** command tests various outflow threshold conditions such as expected wait time. If the expressed condition is met, a branch is made to the appropriate **route-to** command. This command sends the call to the receiving Communication Manager, which, as already noted, can accept or deny the call.

The following example shows an outflow vector that can be included in a sending Communication Manager.

Using LAI with route-to commands to outflow calls

```
1. wait-time 0 secs hearing ringback
2. goto step 5 if expected-wait for split 3 pri m < 30
3. route-to number 5000 with cov n if unconditionally
4. route-to number 95016781234 with cov n if unconditionally
5. queue-to split 3 pri m
6. announcement 3001
7. wait-time 30 secs hearing music
8. goto step 6 if unconditionally
```

If split 3 has an expected wait time of less than 30 seconds (step 2), step 5 queues the call to the queue at a medium priority.

If the expected wait time is more than 30 seconds, LAI attempts are made in steps 3 and 4. If the call is accepted by one of the receiving Communication Manager call control passes to the receiving Communication Manager.

If the receiving Communication Manager deny the call, the call queues to split 3 and announcement 3001 plays. The caller hears music, interrupted by announcement 3001 every 30 seconds.

Receiving switch operation

When the receiving switch receives the LAI request, the call first routes to a VDN. The VDN then maps the call to the receiving switch's inflow vector, and vector processing begins, starting with inflow checking. Inflow checking is enabled by conditional **goto** commands in the inflow vector. The decision to accept or deny a call can be based on checks such as any of the following:

- Expected Wait Time
- Number of staffed agents
- Number of available agents
- Time-of-day or day of the week
- Number of calls in split queue
- Average Speed of Answer
- Active VDN calls
- Automatic Number Identification (ANI) and Calling Line Identification (CLID)
- II-Digits

- CINFO ced or cdpd digits
- Collected digits forwarded from the sending switch

Once inflow checking is complete, acceptance of the LAI call is accomplished by executing any of the vector commands listed in the call-acceptance vector commands.

Note:

For each command listed in the call-acceptance vector commands, neutral vector commands, and call-denial vector commands, only one of the corresponding qualifications must be true for the command to cause the desired result, which is call acceptance, call denial, or no effect on such acceptance or denial.

The following example shows an inflow vector that can be used by a receiving switch.

Using inflow checking for LAI requests

```
1. goto step 6 if expected-wait in split 1 pri h > 30
2. queue-to split 1 pri h
3. announcement 4000
4. wait-time 2 seconds hearing music
5. stop
6. busy
```

Step 1 of this inflow vector checks the inflow thresholds. The **`goto step`** command in step 1 checks the expected wait time in split 1. If the expected wait time is greater than 30 seconds, a branch is made to the **`busy`** command in step 6. If executed, the **`busy`** command denies the call, and the receiving switch returns a call denial message to the sending switch. The sending switch, in turn, drops the LAI call attempt and then continues vector processing at the next vector step.

If the expected wait time in split 1 is less than or equal to 30 seconds, the receiving switch returns a call acceptance message to the sending switch, and call control is passed to the receiving switch. Thereafter, the call is queued to split 1 in the receiving switch (step 2). Once queued, the caller receives the appropriate announcement in step 3 and is then provided with music until the call is answered by an agent or abandoned by the caller (steps 4 and 5). Remember that the **`stop`** command halts vector processing but does not drop the call.

If the sending switch does not receive a call acceptance or call denial message within 120 seconds after the LAI call request, the LAI attempt is dropped. The sending switch continues vector processing at the next step.

Enhanced LAI

Enhanced Look-Ahead Interflow (LAI) uses the same basic Call Vectoring commands as traditional LAI, but includes an **`interflow-qpos`** conditional. Use Enhanced LAI when the preferred call flow performs an LAI attempt after putting a call in a queue.

The **`interflow-qpos`** conditional:

- Produces First in First Out (FIFO) or near FIFO call processing.
- Uses less processing during LAI.

The simple way to achieve FIFO

You can use the *interflow-qpos* conditional in a **route-to** or **goto** command to achieve First In, First Out (FIFO) results.

For example, you can use the following **route-to** command with the conditional to achieve FIFO results:

route-to number 9581234 with cov `n` if interflow-qpos=1

If you have a lot of remote agents, set the **route-to** command as follows:

route-to number 9581234 with cov `n` if interflow-qpos<=2

The interflow-qpos conditional

The **interflow-qpos** conditional only applies interflow processes to a dynamic eligible queue and to calls that are queued locally before the **route-to** is attempted.

The eligible queue is that portion of the split/skill queue that:

- Includes only calls that are not expected to be answered locally during the interflow process at that moment relative to the call being processed
- Does not include direct agent calls because these calls are excluded from any interflow process.

The following is an example of the **interflow-qpos** conditional used in a **route-to** command:

```
route-to number _____ with cov _ if interflow-qpos CM x
```

where

- **CM** is the comparator. It is one of three symbols: =, <, <=
    - With **if interflow-qpos = x**, the call is interflowed if it is at the **x** position from the top of the eligible queue.
    - With **if interflow-qpos < x**, the call is interflowed if it is among the top **x-1** of the eligible queue.
    - With **if interflow-qpos <= x**, the call is interflowed if it is among the top **x** eligible calls.
- **x** indicates the call position in the eligible queue. Valid queue positions are 1 through 9. The top queue position is 1. The eligible queue is made up of calls from the first local split or skill that the call has been queued to due to previous steps in the vector.

Note:

Calls that are likely to be serviced locally before an LAI can be completed are not eligible for interflow since the calls are excluded from the eligible queue. Calls that are likely to be answered are identified based on conditions of the split/skill to which the call is queued and, under certain conditions, an administered minimum EWT threshold value.

The following is an example of the **interflow-qpos** conditional used in a **goto** command:

```
goto step/vector ____ if interflow-qpos CM x
```

where

- **CM** is the comparator. It is one of six symbols: =, <>, <, <=, >, >=
- **x** indicates the call's position in the eligible queue. Valid queue positions are 1 through 9. The top queue position is 1.

Calls that are likely to be serviced locally before an LAI can be completed are not eligible for interflow since the calls are excluded from the eligible queue.

When does a call not interflow?

A call does not interflow under the following circumstances:

- If the interflow-qpos conditional is not met.

  As with other conditionals, the route-to number... if interflow-qpos step or the goto step/vector branch is executed only if the conditional is met, otherwise vector processing goes to the next step.

- If the call is not in a split or skill queue, or not in the eligible portion of the queue when the conditional step is executed.

  If the call is not in queue when the "route-to number... if interflow-qpos" step is executed, a vector event is logged and vector processing continues at the next step.

  If the call is not in queue when a "goto... if interflow-qpos" step is executed, the queue position of the call is treated as infinite in determination of the conditional.

  Note:

  A vector event is not logged if the call is in queue, but is not in the eligible portion of the queue.

- Interflow failure or LAI rejection

  Interflow failure or LAI rejection will also go to the next step. Route-to operation and feature interactions will be the same as other configurations of the route to number command, for example, **route to number ___ with cov _ if digit CM x**.

The following table outlines what action is taken for different cases of interflow eligibility.

Case	Action at route-to step	Action at goto step
The call not eligible for interflow.	The call is never routed.	Treat as if the interflow queue position is infinite.
The call is not in any split queue.	The call is treated as if the interflow queue position is infinite.	Treat as if interflow queue position is infinite.
The call is eligible for interflow.	Act according to the conditional.	Act according to the conditional.

Setting the minimum EWT

About this task

The minimum expected wait time (EWT) threshold that is used to help determine which calls are more likely to be answered locally is administered on the Feature-Related System Parameters screen. Minimum EWT is used when the local agents, that is, in the first split/skill to which the call is queued, are handling a significant number of the calls. If these agents are not handling a significant number of calls, the call is eligible for LAI even if its EWT is lower than the threshold.

Note:

When enhanced LAI vectors or the look-ahead EWT threshold are administered inappropriately, remote agents can experience phantom calls or a delay between becoming available and receiving an ACD call.

Use the SAT terminal or terminal emulator to administer Communication Manager.

To set the minimum EWT threshold:

Procedure

1. In the command line, type `change system-parameters feature` and press `Enter`.

   The system displays the Feature-Related System Parameters screen.

2. Find the page of the Feature-Related System Parameters screen that has the **Interflow-Qpos EWT Threshold** field.

   If Look-Ahead Interflow is active, the **Interflow-Qpos EWT Threshold** field can be administrated.

3. In the **Interflow-Qpos EWT Threshold** field, enter the number of seconds, as a number from 0 to 9, that you want for the EWT threshold.

   The default of two seconds is recommended.

   Note:

   When the **look-ahead EWT threshold** field is set too low, remote agents can experience phantom calls.

4. Press `Enter` to save your changes.

Example of single-queue multisite operation

In the scenario, all new calls for a given customer application are routed by the public network to only one of the switches in the network, where the calls are put in the queue.

Local agents service the calls from the queue in the normal fashion. However, remote agents service calls by means of Enhanced Look-Ahead Interflow (LAI).

The Communication Manager server with the call queue does rapid enhanced look-ahead attempts to all other switches in the network that can service this call type, looking for an available agent.

Normally, the look-ahead attempts are placed only on behalf of the call that is at the head of the queue (**`interflow-qpos = 1`**). However, in scenarios where there are large numbers of agents at a remote Communication Manager, interflows on behalf of more than one call can be necessary to outflow a sufficient volume of calls to keep all agents busy (**`interflow-qpos <= 2`**).

Vector to back up split

```
1. announcement 3501
2. wait-time 0 secs hearing music
3. queue-to skill 1 pri m
4. route-to number 93031234567 with cov n if interflow-qpos = 1
5. route-to number 99089876543 with cov n if interflow-qpos = 1
6. wait-time 5 secs hearing music
7. goto step 4 if unconditionally
```

Interflow call attempts are placed on behalf of the call that is at the beginning of the queue every five seconds to the two other Communication Manager in the network.

If queuing times are very long, five minutes, for example, and the call is not near the beginning of the queue, going through the vector loop from step 4 to step 7 every 5 seconds is of no avail. The FIFO processing vector is more efficient.

Example of maintaining FIFO processing with LAI

One of the advantages of enhanced LAI is the ability to provide FIFO or near-FIFO call processing. The following example shows a vector that is used to achieve such call processing.

FIFO processing vector

```
1. announcement 3501
2. wait-time 0 secs hearing music
3.  queue-to skill 1 pri m
4. goto step 7 if interflow-qpos < 9
5. wait-time 30 secs hearing music
6. goto step 5 if interflow-qpos >= 9
7. route-to number 93031234567 with cov n if interflow-qpos = 1
8. route-to number 99089876543 with cov n if interflow-qpos = 1
9. wait-time 5 secs hearing music
10. goto step 7 if unconditionally
```

In this vector:

- The rapid look-ahead loop is only entered when the call reaches one of the top 8 positions in queue.
- The number of executed vector steps is reduced dramatically when call waiting times are long.

It is important to write vectors so that calls at the head of the queue have advanced to the rapid look-ahead loop by the time their turn to interflow has been reached. In the vector example shown above, if 8 calls can be serviced from queue in less than 30 seconds (which is the loop time on step 5), there can be a delay in outflowing calls to available agents at the remote sites.

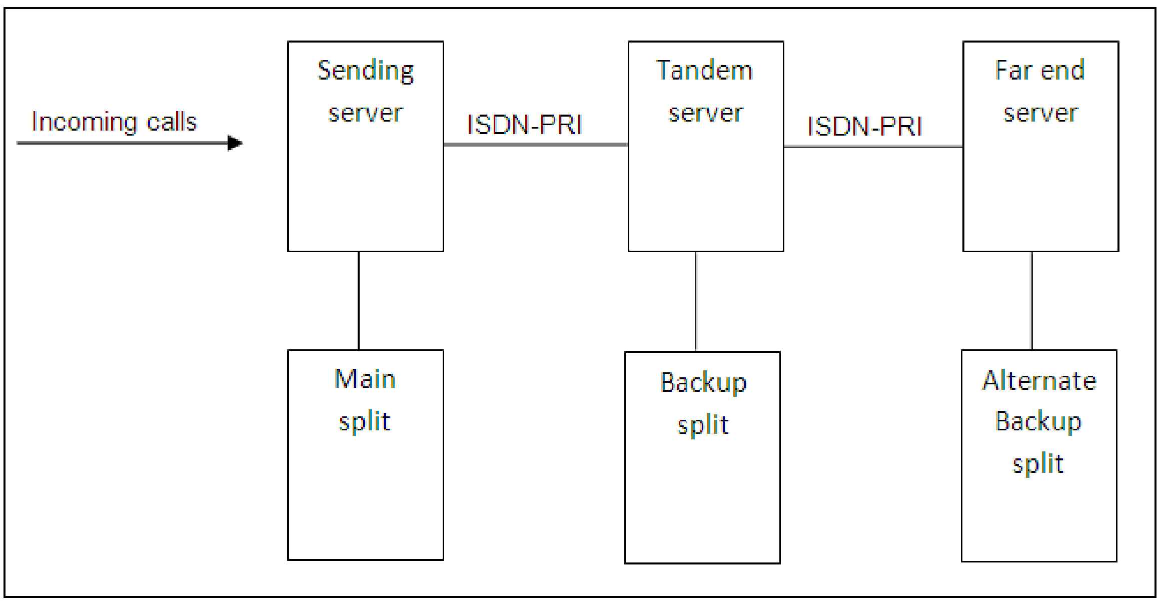
Single-queue FIFO considerations

- When there are available agents, calls are always delivered to available agents at the queuing Communication Manager before available agents at the remote Communication Manager.

- When there are calls in the queue and agents serve calls from multiple applications, the agents always service calls from the applications that are queued locally before calls from applications that are queued at another Communication Manager.
- Use backup VDNs and vectors ito provide continuous operation in the event of a failure at a queuing Communication Manager.
- EWT predictions cannot be made if the split or skill in which the calls are queued has no working agents.
- EWT predictions can be temporarily inaccurate if there are sudden, major changes in the number of working agents in the split or skill in which the calls are queued.

Example of LAI in a tandem switch configuration

To route calls over ISDN facilities, you can implement tandem LAI by using the **`route-to`** command that contain external destinations. This configuration is shown in the following figure.



Sending switch operation

The sending switch is unaware that its LAI call is being tandemed to an alternate switch. The operation of the sending switch in the tandem switch configuration is the same as that in the two-switch configuration.

Tandem switch operation

If the receiving Communication Manager executes a **`route-to`** command that routes the call over an ISDN facility before call acceptance, the **`route-to`** command is performed on a look-ahead basis in the same manner as a sending Communication Manager. If the call is accepted at a far-end Communication Manager, acceptance is passed to the sending Communication Manager, and call control is passed to the far-end Communication Manager, along with tandeming of the original calling party information and the original DNIS name. If the call is denied, the next step of the tandem switch vector is executed.

The following example shows a tandem switch vector.

Tandem switch vector example

```
1. goto step 6 if expected-wait in split 30 pri h > 30
2. queue-to split 30 pri h
3. announcement 200
4. wait-time 2 seconds hearing silence
5. stop
6. route-to number 4000 with cov n if unconditionally
7. busy
```

Step 1 of this vector checks the inflow threshold. If the inflow criteria are acceptable, the vector flow drops to step 2, where the **`queue-to split`** command provides acceptance to the sending Communication Manager. Thereafter, steps 3 through 5 provide a typical queuing-wait scheme.

If the inflow criteria are not acceptable, a branch is made to step 6. The **`route-to`** command in this step checks another Communication Manager that is enabled with LAI on a look-ahead basis. If this far-end Communication Manager rejects the call, a denial message is relayed back to the tandem Communication Manager, which then drops the LAI call attempt. On the other hand, if the far-end Communication Manager accepts the call, an acceptance message is relayed all the way back to the sending Communication Manager.

No ringback is provided in this tandem Communication Manager vector. This is necessary so that an acceptance message is not returned to the sending Communication Manager. This operation is appropriate for the caller because the sending Communication Manager has already returned an announcement before a LAI attempt is made to the receiving Communication Manager.

Be sure that the sending Communication Manager is not used as a backup location for the tandem Communication Manager or for any of the far-end Communication Manager. If the sending Communication Manager is administered in this manner, all trunk facilities can be tied up by a single call.

Far-end switch operation

The far-end switch is also unaware that tandeming has taken place. The far-end switch functions in the same manner as the receiving switch within the two-switch configuration.

LAI-initiated path replacement for calls in vector processing

Path replacement for calls in vector processing

You can use QSIG or DCS with Reroute using ISDN SSE for path replacement for calls in a queue. For calls that are waiting in a queue or in vector processing, even if the call is not connected to an answering user, Communication Manager can attempt a path replacement to find the optimum path for the call.

You can use the **`route-to`** command with LAI to initiate a QSIG path replacement for a call.

If the terminating Communication Manager server receives a *Path Replacement Propose* operation for a call that is in a queue or in vector processing, Communication Manager immediately initiates path replacement using the path replacement extension if the **Path Replace While in Queue/Vectoring** field is `y` and the **Path Replacement Extension** field has a valid entry. The fields are available on the ISDN Parameters page of the Feature-Related System Parameters screen.

Example LAI vector

The following example shows how an LAI vector can be written to trigger path-replacement at the terminating switch.

Note:

In order for a path-replacement to be attempted, the incoming and outgoing trunks that are used for the call must be administered with the **Supplementary Service Protocol** field set to `b`.

LAI-initiated path-replacement vector

```
1. wait 0 seconds hearing music
2. queue-to skill "n" if available-agents < 6
3. route-to number "ARS number for ISDN trunk" with cov n
4. wait 999 seconds hearing ringback
```

At the receiving Communication Manager, the vector that processes the incoming call must use an **`announcement`**, or **`wait hearing music`** vector command to enable path-replacement.

DNIS and VDN Override in an LAI environment

LAI handles Dialed Number Identification Service (DNIS) and VDN Override in various ways, depending on a number of different characteristics of the call. DNIS, as described in Call Vectoring fundamentals, allows any agent with a display-equipped telephone to receive visual displays that specify the name of the called VDN. VDN Override in its basic form allows the name of a subsequently routed to VDN to be displayed to the answering agent instead of the name of the originally called VDN.

The following sections discuss how LAI handles DNIS and VDN Override.

DNIS information displayed to answering agent

For LAI, the DNIS name, which is the called VDN name from the sending switch, is presented on the display for the answering agent on the receiving switch if all of the following are true:

- The LAI option is enabled.
- The call routes to a VDN.
- The DNIS name field is not blank.

The type of DNIS information that is displayed depends upon a number of different scenarios. This information is presented in the following table.

Scenario	Information displayed
Tandem LAI call	Look-Ahead Interflow DNIS information from the original LAI call.
No redirection at the sending switch	VDN name according to Override rules at the sending switch (active VDN).

Table continues…

Scenario	Information displayed
Redirection at the sending switch (VDN in coverage path)	Original VDN name, or If multiple VDNs are accessed, the name of the VDN that was last accessed by a `route-to` command.
Sending switch sends a blank DNIS Name field (that is, a name is not assigned to the sending switch called VDN) or the trunk group is administered to not send the LAI name.	Name associated with the receiving VDN. This name can be changed according to the rules of VDN Override at the receiving switch.

Note:

VDNs that map to vectors that place LAI calls must have their ISDN Calling Party Number (CPN) prefixes administered. If an ISDN CPN prefix is not administered, the assigned VDN name is not sent. Instead, a DNIS of all blank space characters is sent and displayed on the answering agent's terminal.

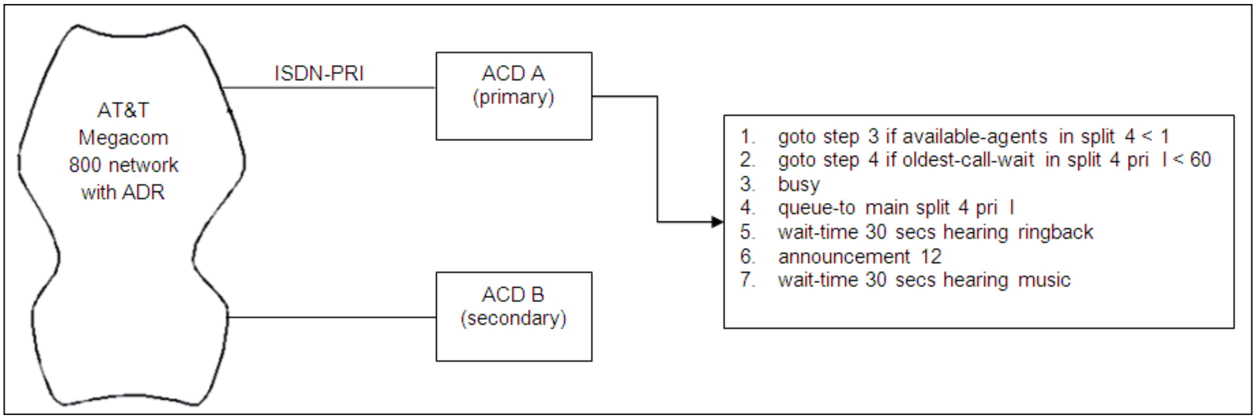
Originator display

For internal calls, the originator display contains the same information as for Basic Call Vectoring, but the originator can receive unwanted display updates during LAI call attempts. In this case, LAI calls must go out over trunk groups that have the **Outgoing Display** field set to `n` and internal callers who call the trunk group can view the dialed digits on their phone display.

LAI with network ADR

Call Vectoring and LAI are compatible with and supplement the network services Alternate Destination Redirection (ADR) rerouting feature or equivalent service from other network providers. ADR uses ISDN-PRI connectivity with the switch in the same manner as LAI to allow the receiving system to indicate whether a call is to be accepted or rejected. The same type of vector that is used as a receiving ACD for LAI is used at the ADR-receiving ACD. If the call is accepted, it is connected to the system. If the call is rejected, the network routing number is translated to another number that routes the call to the alternate location within dialing-plan constraints. ADR allows for only one alternate location. LAI can be used at the alternate location to test other locations for less-busy conditions.

The following figure shows the configuration for a multilocation application.



The network requires ISDN-PRI connectivity to primary location A. Connection to secondary location B does not have to be ISDN-PRI. ADR attempts to route the call to location A over the ISDN-PRI link using a routing number that selects a VDN that is assigned to the receiving vector shown.

When the routing attempt is made, Call Vectoring starts processing the vector. The example then proceeds at location A as follows:

1. Step 1 checks for staffing of the ACD split, and branches to step 3 if it is not staffed.
2. If the ACD split is staffed, step 2 checks the oldest call waiting time in the split, and branches to step 4 if it is less than 60 seconds.
3. If the ACD split is unstaffed or if the oldest call waiting time is more than 60 seconds, step 3 rejects the call and returns a busy indication to the network.
4. If the oldest call waiting time is less than 60 seconds, step 4 accepts the call and queues it. ADR then connects the call through to the receiving system.
5. Steps 5 through 7 provide ringback, announcement, and music to the caller.

If the vector at location A rejects the call by sending a busy indication back to the network over the ISDN-PRI link, ADR reroutes the call to location B which must accept the call. If location B is closed or too busy to take the call, location B can use Call Vectoring and LAI to check other locations. If other locations exist and can take the call, location B can forward the call. If other locations do not exist or cannot take the call, location B can use Call Vectoring to route the call to location A. If location A is not open, location B can use Call Vectoring to provide an announcement or a busy tone to the caller.

Multisite applications for Enhanced LAI

Enhanced LAI has the following applications in a multisite environment:

- You can implement a single queue First-In First-Out (FIFO) operation for any application. However, for maximum efficiency and flexibility, use BSR instead of LAI.
- You can use LAI with BSR in networks with extremely low call volumes.

Meet-me Conference

The Meet-me Conference feature allows you to set up a dial-in conference of up to six parties. The Meet-me Conference feature uses Call Vectoring to process the setup of the conference call.

Meet-me Conference can be optionally assigned to require an access code. If an access code is assigned, and if the vector is programmed to expect an access code, each user dialing in to the conference call must enter the correct access code to be added to the call.

The Meet-me Conference extension can be dialed by any internal or remote access users, and by external parties if the extension number is part of the customer's DID block.

Meet-me Conference command set

The following table lists the commands associated with Meet-me Conference.

Command category	Action taken	Command
Information collection	Collect information from the calling party.	**collect digits**
Treatment	Play an announcement.	**announcement**
	Play a busy tone and stop vector processing.	**busy**
	Disconnect the call.	**disconnect**
	Delay with audible feedback of silence, ringback, system music, or alternate audio or music source.	**wait-time**
Routing	Route to the appropriate meet-me conference and stop vector processing.	**route-to**
Branching or programming	Go to a vector step.	**goto step**
	Go to another vector.	**goto vector**
	Stop vector processing.	**stop**

Information collection commands

collect command

Syntax

`collect 6 digits after announcement <extension>`

When the **Meet-me Conf** field is set to `y`, the collect vector step has been modified to collect the next six digits and use those digits as the access code for a Meet-me Conference call. Though not required, the digits can be collected after a recorded announcement.

Treatment commands

announcement command usage for Meet-me Conference

Syntax

`announcement <extension>`

The usage for the **announcement** command is the same as in Basic Call Vectoring. For details on using this command, see the Basic Call Vectoring section.

busy command usage for Meet-me Conference

Syntax

```
busy
```

The usage for the **busy** command is the same as in Basic Call Vectoring. For details on using this command, see the Basic Call Vectoring section.

disconnect command usage for Meet-me Conference

Syntax

```
disconnect after announcement <extension>
```

The usage for the **disconnect** command is the same as in Basic Call Vectoring. For details on using this command, see the Basic Call Vectoring section.

wait-time command usage for Meet-me Conference

Syntax

```
wait-time <time> secs hearing <silence, ringback, music>
```

The usage for the **wait-time** command is the same as in Basic Call Vectoring.

Routing commands

The following section details the syntax that can be used for this command and any information that is specific to the Meet-me Conference feature.

route-to meetme command

Syntax

```
route-to meetme
```

The route-to vector step uses the condition meetme only for the Meet-me Conference feature. When successful, this condition adds the caller to the Meet-me Conference call and all parties on the call hear an entry tone to signify that another caller has joined the conference. This condition is valid when the caller has entered the correct access code and there are not already six parties on the call.

If the route to meetme step ever fails, vector processing stops and the caller hears busy tone.

Branching/programming commands for Meet-me Conference

Meet-me Conference uses several branching/ programming commands, including:

- goto step
- stop

The following sections detail the syntax that can be used for these commands and any information that is specific to their use in Attendant Vectoring.

goto step command for Meet-me Conference

Syntax 1

```
goto step <step #> if meet-me-idle
```

Syntax 2

```
goto step <step #> if meet-me-full
```

The goto step vector step has two conditions used for the Meet-me Conference feature:

- meet-me-idle
- meet-me-full

The meet-me-idle condition routes the first caller accessing a Meet-me Conference to the conference call. An announcement step informs the caller that the caller is the first party to access the call.

The meet-me-full condition is used when the Meet-me Conference already has the maximum of six parties on the call.

Syntax 3

```
goto step <step #> if digits = meet-me-access
```

The goto step vector step supports the option, meet-me access, for the digits condition to verify that the access code is valid. If the access code entered by the caller equals the access code administered for the VDN, vector processing continues.

stop command for Meet-me Conference

The use of the **`stop`** command is the same as in Basic Call Vectoring. For details on using this command, see the Basic Call Vectoring section.

Activating Meet-me Conference

About this task

Meet-me Conference is available for all Communication Manager releases that support Avaya Call Center R11 or later.

To enable the Meet-me Conference feature:

Procedure

1. Set the **G3 Version** field on the System-Parameters Customer-Options screen to V11 or later.
2. Set the **Enhanced Conferencing** field on the System-Parameters Customer-Options screen to `y`.

   Enhanced Conferencing has an Right-to-Use (RTU) cost and must be enabled through the License File process.

Creating a Meet-me Conference VDN

About this task

To create a Meet-me Conference VDN (using example VDN 36090):

Procedure

1. Enter the `add vdn 36090`. The following VDN screen appears:

```
add vdn 36090                                               Page   1 of   3   SPE A
                            VECTOR DIRECTORY NUMBER

                             Extension: 36090
                             Name: Enhanced Conf. Meet-me VDN
Vector Number: 90

                    Meet-me Conferencing? y
```

2. Enter a name, a vector number, and enter `y` in the **Meet-me Conferencing** field.
3. Press **NEXTPAGE** to display page 2.

   The system displays page 2 of the VDN screen:

```
add vdn 36090                                            Page   2 of   3   SPE A
                            VECTOR DIRECTORY NUMBER

                         MEET-ME CONFERENCE PARAMETERS
       Conference Access Code: 937821
        Conference Controller: 80378
              Conference Type: 6-party
              Route-to Number:
```

4. Enter a conference access code.

   If you do not want an access code, leave the field blank. Once an access code is assigned, an asterisk displays in this field for subsequent change, display, or remove operations by all users except the “init” super user login.

   Security alert:

   Always assign an access code to a Meet-me Conference VDN.

5. Enter a conference controller extension.

   If an extension number is entered, a user at that extension can change the access code for the Meet-me Conference VDN using a feature access code. If this field is blank, only a station user that is assigned with console permissions can change the access code for the Meet-me Conference VDN using a feature access code. In addition, remote access users can change a Meet-me Conference access code using the feature access code.

6. Enter the conference type.

   This field can have the following values:

   - 6-party - Enter this value to administer a regular 6-party conference. This value is the default.
   - expanded - Enter this value to administer up to a 300-party conference.

7. If you set the **Conference Type** field to expanded, use the **Route-to Number** field to administer the ARS/AAR Feature Access Code, the routing digits, and the conference ID digits for the VDN.
8. Press **Enter** to submit the VDN.

Creating a Meet-me Conference vector

About this task

To create a Meet-me Conference vector (using example vector number 90):

Procedure

1. Enter:

   `change vector 90`

   The system displays the CALL VECTOR screen.
2. Enter **y** in the **Meet-me Conf** field.

   This designates the vector as a Meet-me Conference vector.
3. Create a vector as shown in the following example:

```
change vector 90                                        Page   1 of   3   SPE A
                                  CALL VECTOR

    Number: 90               Name: Meet-me Vec
Multimedia? n      Attendant Vectoring? n    Meet-me Conf? y           Lock? y
     Basic? y   EAS? n   G3V4 Enhanced? n   ANI/II-Digits? n   ASAI Routing? n
 Prompting? y   LAI? n  G3V4 Adv Route? n   CINFO? n   BSR? n   Holidays? n

01 collect      6    digits after announcement 12340
02 goto         step   6   if digits           =      meet-me-access
03 collect      6    digits after announcement 12341
04 goto         step   6   if digits           =      meet-me-access
05 disconnect   after announcement 12342
06 goto         step   11  if meet-me-idle
07 goto         step   14  if meet-me-full
08 announcement 12343
09 route-to     meetme
10 stop
11 announcement 12344
```

```
change vector 90                                        Page   2 of   3   SPE A
                                  CALL VECTOR

12 route-to     meetme
13 stop
14 disconnect   after announcement 12345
15 stop
16
17
18
19
20
21
22
```

4. Press ENTER to submit the vector.

Meet-me Conference capacity issues

Meet-me Conference calls count towards the maximum number of 3-way and 6-way conference calls.

Users cannot add more parties to a conference call once the system maximum is reached.

For Category A, the number of Meet-me Conference VDNs is a subset of the total number of VDNs allowed in the system.

For Category B, the total number of VDNs and vectors is doubled from the normal limit if both Call Vectoring and Enhanced Conferencing are enabled. However, the maximum number of VDNs and vectors available for call center applications is unchanged.

Meet-me Conference call processing scenario

Joe Davis has a sales review scheduled with four associates located in different cities. He has reserved Meet-me Conference telephone number 865-253-6090. In switch administration, this number has been assigned to vector 90. See the following screen.

```
add vdn 36090                                            Page   1 of  3   SPE A
                            VECTOR DIRECTORY NUMBER

                             Extension: 36090
                                  Name: Meet-me VDN                        Vector
Number: 90

                    Meet-me Conference? y
```

VDN 36090 is administered with an access code of 835944.

When each associate calls the Meet-me Conference telephone number, the following vector processing occurs:

```
change vector 90                                         Page   1 of   3   SPE A
                                  CALL VECTOR

    Number: 90               Name: Meet-me Vec
                   Attendant Vectoring? n    Meet-me Conf? y           Lock? y
     Basic? y   EAS? n   G3V4 Enhanced? n   ANI/II-Digits? n   ASAI Routing? n
 Prompting? y   LAI? n  G3V4 Adv Route? n   CINFO? n   BSR? n   Holidays? n

01 collect      6    digits after announcement 12340
02 goto         step   6   if digits          =      meet-me-access
03 collect      6    digits after announcement 12341
04 goto         step   6   if digits          =      meet-me-access
05 disconnect   after announcement 12342
06 goto         step   11  if meet-me-idle
07 goto         step   14  if meet-me-full
08 announcement 12343
09 route-to     meetme
```

```
10 stop
11 announcement 12344
```

```
change vector 90                                          Page   2 of   3   SPE A
                                   CALL VECTOR
12 route-to     meetme
13 stop
14 disconnect   after announcement 12345
15 stop
16
17
18
19
20
21
22
```

Each caller hears announcement 12340, which says something similar to “Welcome to the Meet-me Conferencing service. Enter your conference access code.” Each caller enters the access code 835944.

The **`collect`** vector step 1 collects the access code digits. If the access code is valid, vector processing continues with vector step 6. If the access code is invalid, vector processing continues with vector step 3, which plays announcement 12341. Announcement 12341 says something similar to “This access code is invalid. Please enter the access code again.” If the caller enters the wrong access code again, vector processing continues with vector step 5, which plays announcement 12342. Announcement 12342 says something similar to “This access code is invalid. Please contact the conference call coordinator to make sure you have the correct conference telephone number and access code. Goodbye.”

Vector step 6 is only valid for the first caller into the Meet-me Conference. The meet-me-idle condition routes the first caller to announcement 12344 (vector step 11). The recorded announcement says something similar to, “You are the first party to join the call.” The caller is then routed to the Meet-me Conference call by vector step 12 and vector processing stops.

Vector step 7 is used when the Meet-me Conference already has the maximum of six parties on the call. The meet-me-full condition disconnects the caller after playing announcement 12345 (vector step 14). The recorded announcement says something similar to, “This Meet-me Conference is filled to capacity. Please contact the conference call coordinator for assistance. Goodbye.”

If a caller enters the correct access code, is not the first caller, and the conference call is not full, vector processing continues with vector step 8, which plays announcement 12343. The announcement says something similar to “Your conference call is already in progress.” The caller is then routed to the Meet-me Conference call by vector step 9 and vector processing stops. As each caller enters the conference call, all parties on the call hear an entry tone.

When the conference call is over and callers drop out of the conference call, any remaining parties on the call hear an exit tone.

Meet-me Conference interactions

Both Attendant Vectoring and Meet-me Conference cannot be enabled at the same time.

If **Enhanced Conferencing** is enabled, but no other vectoring customer options are enabled, only Meet-me Conference vectors can be assigned.

A non Meet-me Conference vector cannot be assigned to a Meet-me Conference VDN and a Meet-me Conference vector cannot be assigned to a non Meet-me Conference VDN.

There will be no restrictions in vector chaining between Meet-me Conference and non Meet-me Conference vectors (for example, using the **`goto vector`** or **`route-to number`** commands. When calls interflow from one type of vector processing to another, the calls will be removed from any queue and treated as new calls to vectoring, not a continuation of vectoring.

Call Detail Recording

As parties join a Meet-me Conference, a call record is created if required by system administration. In the call record, the called party is the Meet-me Conference VDN number and the duration is the length of time that the party is included in the call. An individual record for each party is generated when the party drops from the call. One option that records all calls to Meet-me Conference VDNs is to activate the Intra-switch CDR feature and populate all the Meet-me Conference VDN numbers in the system.

If Intra-switch CDR is used with the Meet-me Conference VDNs, the condition code must be set to `C` for all call records as is done with traditional conference calls when Intra-switch CDR is active.

If Intra-switch CDR is not enabled for Meet-me Conference VDNs, the creation and contents of call records depends on the trunk group translations for external callers to the Meet-me Conference. For internal callers to the Meet-me Conference, no record is generated if the Intra-switch CDR feature is not active for either the Meet-me Conference VDN or the calling extension.

Changing vector types

To change a Meet-me Conference vector to a non Meet-me Conference vector and vice versa, you must first remove all vector steps. If either of the conditions exists, a warning message displays stating the VDNs currently assigned to the vector is not operating as expected. The next time you try to submit a change to the Meet-me Conference VDN, you have to assign the VDN to a Meet-me Conference vector.

Direct Inward Dialing

If the VDN extension is part of the customer DID block, external users can access the conference VDN. If the VDN extension is not part of the customer DID block, only internal callers on the customer network, including DCS or QSIG, or remote access callers can access the conference VDN.

Disabling Enhanced Conferencing

If Meet-me Conference VDNs are assigned when disabling the **Enhanced Conferencing** field, the change is not allowed and the message, “Must first remove all Meet-me Conf VDNs and vectors ” appears. You must remove the VDNs and vectors before you set the **Enhanced Conferencing** field to `n`.

Removing stations

A station that is administered as a controlling station for a Meet-me Conference VDN cannot be removed without first removing the assignment on the VDN. The following message displays:

```
Must first remove as conference controller on VDN form.
```

Multi-National Calling Party Number prefixes

The Calling Party Number (CPN) that accompanies a call can be used for a variety of purposes, including phone display, routing, billing, and screen pops using ASAI. However, using the CPN to identify the origin of the call can produce ambiguous results. For instance, the country code for Germany and the city code for Padova, Italy, are the same (49), so calls from the two locations will each begin with the same digits. The prefix in a multinational CPN helps differentiate local or national numbers from international numbers, allowing Communication Manager to identify calls as national (NTL) or international (INTL). For example, in the U.S. the national prefix is 1 and the international prefix is 011, whereas in Europe the digits are typically 0 and 00. When the values are administered in Communication Manager, the values are included as part of the CPN and can be displayed on an agent's phone or passed to ASAI so that the call can be handled appropriately.

A feature is available on the System-Parameters Feature-Related screen to optionally pass the CPN prefix to VDNs and vectors as part of the CPN. You can use this option on a system-wide basis, or on an individual VDN screen. Accordingly, international call handling can be improved by prioritizing and routing more intelligently such as by automatically routing calls to agents who speak the language. For example, if a customer from Germany contacts to a call center in Spain, based on the international code of Germany (0049 - international prefix + country code), the call can be routed to a German speaking agent. Also, an international call can be given priority over a national call to save toll charges.

CPN prefix routing example

In the following example Vector Routing table includes the country codes for countries in South America, including the US international CPN prefix.

```
VECTOR ROUTING TABLE
      Number: 10      Name: South America       Sort? n
           1: 01154                               17:
           2: 011591                              18:
           3: 01155                               19:
           4: 01156                               20:
           5: 01157                               21:
           6: 011593                              22:
           7: 011594                              23:
           8: 011592                              24:
           9: 011595                              25:
          10: 01151                               26:
          11: 011598                              27:
          12: 01158                               28:
          13: 011597                              29:
          14: 011500                              30:
```

```
         15:                                      31:
         16:                                      32:
```

The following vector example shows how calls are routed based on the administered international CPN prefix, country codes, and the information provided in the Vector Routing Table shown above.

```
1. wait-time 0 seconds hearing ringback
2. goto step 11 if ani = none
3. goto vector 1910 @step 1 if ani = 01149+
4. goto vector 1912 @step 1 if ani in table 10
5. goto vector 1915 @step 1 if ani = 01152+
6. goto vector 1920 @step 1 if ani = 011+
7. queue-to split 16 pri m
8. wait-time 60 seconds hearing 32567 then music
9. announcement 32456
10. goto step 8 if unconditionally
11. route-to number 0 with cov n if unconditionally
12. busy
```

In the above example:

- Step 2 checks for calls that do not have an associated ANI, and routes such calls to an operator.
- Step 3 routes calls from Germany to a vector 1910.
- Step 4 routes calls from South American countries, where the country codes are listed in the Vector Routing Table 10, to vector 1912.
- Step 5 routes calls from Mexico to vector 1915.
- Step 6 routes all other international calls to vector 1920. Calls that are not rerouted by the previous steps are then queued.

Network Call Redirection

Network Call Redirection (NCR) provides a call routing method between sites on a public network or a Virtual Private Network (VPN) reducing trunking costs. Cost reductions are particularly valuable in enterprises or multisite call center environments where trunk costs are high.

Public Switch Telephony Networks (PSTNs) or VPNs manage call redirection when an incoming call arrives at Communication Manager that has the NCR feature enabled. As a result, trunks that the server normally retains to accomplish a necessary trunk-to-trunk transfer are released after call redirection takes place.

Cost reductions associated with reduced trunk use can be significant, particularly when you implement Avaya virtual routing features such as Best Service Routing (BSR) with Expected Wait Time (EWT). You can achieve cost savings by using fewer trunks to handle the same number of incoming or outgoing calls after implementing NCR within the local Communication Manager.

NCR options supported by PSTNs

This section describes the various NCR redirection options that are supported by Public Switched Telephone Networks (PSTNs). All the NCR protocols described in this section support Information

Forwarding using User-to-User (UUI) transport to the redirected-to location over the PSTN or VPN network.

Protocols not supported by NCR

The PSTN call-redirection protocols that are currently available but are *not* supported by the NCR feature include the following:

- AT&T 4ESS Out-of-Band Transfer and Connect
- Nortel Release Link Trunks (RLT)

NCT-type feature operations

A key advantage of Network Call Transfer (NCT)-type protocols invoked by Call Vectoring or by manual station call-transfer or call-conference/release operations is that the redirecting server retains control over the call and can continue to use a trunk-to-trunk connection if the PSTN switch does not accept the request to merge B-channels for both legs of the call. If the PSTN switch returns a `failure` message to the PSTN, the originating server maintains a trunk-to-trunk connection for the call. For vector call processing, NCR call processing still treats the NCR attempt to be successful, but the following outcomes occur:

- A vector event is logged to indicate that the NCT operation as attempted with the PSTN failed.
- If CMS is used to track incoming calls to an externally measured VDN, the call is not counted as deflected.

For NCR invocation by Call Vectoring, the local Communication Manager sets up the second leg of the call, waits for the second site to be connected, and requests the PSTN or VPN switch to merge the first leg of the call with the second leg. If this request is accepted, the PSTN or VPN switch joins the original caller to the redirected-to endpoint, sends a `success` message back to the redirecting server and then drops both legs of the call at the redirecting server.

For NCR NCT or TBCT invocation by a station, ACD agent, VRU, or CTI-controlled call doing a call-transfer or call-conference/release operation, if the second leg of the call is set up over an outgoing trunk in the same signaling group as the incoming call, then call-redirection takes place when the call-transfer or call-conference/release occurs. For the NCR ISDN ETSI ECT protocol, the call redirection will take place when the outgoing trunk B-channel either has the same or a different D-channel than the incoming call.

Specific NCT-type protocols

Specific NCT-type protocols include the following protocols:

MCI Network Call Transfer over ISDN PRI

Network Call Redirection and PSTN switch operations associated with the MCI NCT protocol are consistent with those described in NCT-type feature operations on page 153.

MCI Network Call Redirection/Network Call Transfer is compliant with ANSI Explicit Network Call Transfer (ENCT) T1.643 (1995), the MCI Nortel variant of ANSI ECT (1995).

Note:

MCI NCT is offered in the United States by MCI for their Nortel DMS-250 and Alcatel DEX-600 PSTN switches.

Two B-Channel Transfer (TBCT) over ISDN PRI

Network Call Redirection and PSTN switch operations associated with the TBCT protocol are consistent with those described in NCT-type feature operations on page 153.

The Network Call Redirection/Telcordia Two B-Channel Transfer (TBCT) protocol is compliant with the Telcordia Two B-Channel Transfer and ANSI Explicit Call Transfer (1998) standards. For more information, see any of the following:

- Telcordia GR-2865-CORE
- ANSI T1.643 (1998)
- Lucent 99-5E-7268

Note:

TBCT is offered in the U.S. by AT&T for their DMS-100 PSTN switches configured with the NI2 network protocol. TBCT is offered in Canada by Bell/Canada for their DMS-100 PSTN switches and by AT&T/Canada, for their DMS-500 PSTN switches.

ETSI Explicit Call Transfer

Network Call Redirection and PSTN switch operations associated with the European Telecommunications Standard Institute (ETSI) Explicit Call Transfer (ECT) protocol are consistent with those described in NCT-type feature operations on page 153.

The Network Call Redirection/ETSI Explicit Call Transfer protocol is compliant with ETSI standard EN 300 369-1.

Note:

ETSI ECT is offered in Europe by France Telecom and other in-country PSTN service providers for their Ericsson AXE-10 PSTN switches. ETSI ECT is offered in the United Kingdom by the company initially known as MCI for their DMS-100 PSTN switches.

Selection of outbound call leg for NCT-type NCR protocols

For the MCI NCT and Telcordia TBCT NCR protocols, the PSTN switch requires that the outbound call leg of a redirected PRI call is in the same trunk group and has the same Direct Access Line (DAL) D-channel as the inbound call. For vector-initiated invocation of the NCR feature by either a BSR queue-to best or route-to number vector step, the Avaya Communication Manager enforces this requirement by automatically selecting an outbound B-channel that has the same signaling group as the incoming call's D-channel. This results in sending the NCR invocation request on the same D-channel used for the first call leg's associated signaling or for the same associated D-channel when the Non-Facility Associated Signaling (NFAS) D-channel backup configuration is used.

For the ETSI ECT NCR protocol, there is no restriction that the outbound PRI call leg must have the same Direct Access Line (DAL) D-channel used for the first call leg's associated signaling. If the PRI trunk group has more than one associated D-channel, NCR processing sets up the second call leg for call redirection using any B-channel in the trunk group independent of its associated D-channel.

NCD

The Network Call Deflection (NCD) operation by a PSTN switch can occur only if the incoming call to Communication Manager is not answered, that is, an ISDN CONNECT message is not sent to the PSTN switch from the incoming server.

NCD is compliant with ETSI Supplementary Services Network Call Deflection ETS 300 207-1 (partial call rerouting in the public network).

Important:

Some Call Vectoring commands cause CONNECT messages to be sent to the PSTN switch. If Call Vectoring methods are used to implement NCR and the PSTN switch supports the NCD protocol, call vectors used to invoke NCR must not include any of the following vector commands:

- **`announcement`**
- **`collect x digits`**
- **`converse-on split/skill`**
- **`wait hearing music`**
- **`wait hearing (announcement extension) then (continue, music, ringback or silence)`**

When Communication Manager invokes NCD, the PSTN switch sets up the second leg of the call instead of the redirecting Communication Manager. There are two PSTN options for NCD specified by the ETSI standards: *retain call until alerting/connect* and *clear call upon invocation*. The *clear call upon invocation* option is commonly referred to as a *partial call reroute*.

When the *clear call on invocation* option is used, a successful NCR/NCD attempt is indicated when the PSTN or VPN switch has validated the NCR request and sends a *call reroute return* DISCONNECT message to the originating server. In this case, the server loses control of the call after it is transferred to the PSTN or VPN redirection endpoint, and no alternate transfer method is possible if the PSTN or VPN switch fails to transfer the call to the second location.

The *retain call until alerting/connect* option is not available because there are presently no known PTSN or VPN offers with this protocol. With this option, the PSTN or VPN switch sets up the second leg of the call, waits until an ALERTING message is received, and then sends a *call reroute return* FACILITY message followed by a DISCONNECT message to the originating server. In this case, if the second leg of the call fails, the server can redirect the call with a trunk-to-trunk connection so that the call is not lost.

NCD is offered in Europe by British Telecom for their Marconi/Plessey System X and Ericsson AXE10 PSTN switches and by Deutsche Telecom for their Siemens EWSD and Alcatel S12 PSTN switches. NCD is offered in Australia by Telstra for their Alcatel S12 PSTN switches.

Reserving trunk group B-channels for NCT-type redirection operations

Administer the Trunk groups used for NCR NCT-type operations as Call-By-Call (CBC) trunk types. Since CBC trunk groups can carry both incoming and outgoing call traffic, situations can occur in which transient levels of incoming traffic occupy all available B-channels. When no B-channels are available for outgoing calls, attempts to set up the outgoing leg for a redirected call fail and the call redirection fails.

Important:

When the NCR feature is used with high volumes of incoming calls, reserve a minimum number of trunk members for the outgoing leg of redirected calls by using CBC Trunk Group Allocation administration. However, the optimum number of trunk members to reserve depends on traffic patterns that are specific to each call center. Call traffic analysis determines if reservation of B-channels is necessary.

If a trunk group has multiple D-channel signaling groups, the CBC Trunk Group Allocation operation does not guarantee the reservation of outgoing trunks associated with a particular D-channel. Instead, the operation reserves outgoing trunks considering the entire trunk group. Therefore, when NCR is attempted for a trunk group having multiple D-channels, the CBC Trunk Group Allocation operation does not always prevent the blockage of the NCR second call leg setup due to unavailable outgoing trunk B-channels.

To reduce the blockage of NCR NCT-type operations due to no available outbound trunk B-channels:

About network facility types

Before you allocate a minimum number of trunk group members to the outgoing legs of NCR calls, you must administer more than one ISDN services or features. The Network Facilities screen includes two predefined features and ten predefined services. The predefined entries are associated with either Network Specific Facilities (NSF) Type 0 or Type 1. You can administer additional user-defined services or features on the Network Facilities screen. User-defined facilities can be Type 0, 1, 2, or 3. You must obtain support agreements with your PSTN service provider for Type 0 or Type 1 facilities.

Type 2 (incoming) and Type 3 (outgoing) facilities do not use NSF codings or require special support by the PSTN. The network facility types are offered because NSF information is not included with ISDN calls in some regions of the world.

Important:

If PSTN does not support NSF, you must specify a Type 3 facility when you reserve trunk members for NCR operations and enable the Usage Allocation Enhancements Optional feature before you can administer a Type 3 facility.

Example trunk allocation for PSTNs that supports NSF codings

The following example of the Network Facilities screen includes the basic default predefined services and features.

```
change isdn network-facilities                                  Page   1 of   2
                                NETWORK-FACILITIES
                        Facility                              Facility
         Name          Type  Coding            Name          Type  Coding
```

```
    sub-operator         0    00110       mega800             1    00010
    operator             0    00101       megacom             1    00011
    outwats-bnd          1    00001       inwats              1    00100
    sdn                  1    00001       wats-max-bnd        1    00101
    accunet              1    00110       lds                 1    00111
    i800                 1    01000       multiquest          1    10000
    ___________          _    _____
    ___________          _    _____
    ___________          _    _____
    ___________          _    _____
    ___________          _    _____
    ___________          _    _____
```

Once you specify the network facilities, you can allocate trunk members on the basis of specific facilities. The following example shows a CBC Trunk Group Allocation screen for a CBC trunk group for which a minimum of one B-channel is always available for the outgoing legs of redirected calls when the mega800 service is used. The specific feature or service that you specify in this screen depends on the support provided by the PSTN.

```
change trunk-group 29
                                     CBC TRUNK GROUP ALLOCATION

Usage Allocation Plan 1        Usage Allocation Plan 2        Usage Allocation Plan 3

                Min# Max#                         Min# Max#                         Min#
Max#
Service/Feature Chan Chan       Service/Feature Chan Chan       Service/Feature Chan
Chan
mega800           1   99
```

Example trunk allocation for PSTNs that do not support NSF codings

The following example of the Network Facilities screen includes the basic default predefined services and features and an additional user-defined, Type 3 (outgoing) feature (bsr-redirect).

```
change isdn network-facilities                                 Page   1 of   2
                                NETWORK-FACILITIES
                        Facility                             Facility
         Name          Type  Coding           Name          Type  Coding
    sub-operator         0    00110       mega800             1    00010
    operator             0    00101       megacom             1    00011
    outwats-bnd          1    00001       inwats              1    00100
    sdn                  1    00001       wats-max-bnd        1    00101
    accunet              1    00110       lds                 1    00111
    i800                 1    01000       multiquest          1    10000
    bsr-redirect         3    _____
    ___________          _    _____
    ___________          _    _____
    ___________          _    _____
    ___________          _    _____
    ___________          _    _____
```

After administering the user-defined feature, you can specify a minimum number of reserved trunk channels to remain available for the outgoing legs of redirected calls when the feature is used.

```
change trunk-group 42
                                     CBC TRUNK GROUP ALLOCATION

Usage Allocation Plan 1        Usage Allocation Plan 2        Usage Allocation Plan 3

                Min# Max#                         Min# Max#                         Min#
Max#
Service/Feature Chan Chan       Service/Feature Chan Chan       Service/Feature Chan
```

```
Chan
bsr-redirect      5   25
```

AT&T In-Band Transfer Connect service

This section describes PSTN redirection operations associated with the AT&T In-Band Transfer Connect℠ service. Details of the service are described in "AT&T Technical Reference 50075" on the AT&T Web site.

NCR provides Information Forwarding support for the AT&T In-Band Transfer Connect℠ network service ISDN D-channel data-forwarding capability. The Information Forwarding feature forwards the UUI that is associated with the call to the redirected-to location. When Call Vectoring and AT&T In-Band Transfer Connect℠ are used to transfer a call and NCR is enabled for the system, the **`disconnect`** vector step causes UUI IE information to be inserted into the ISDN DISCONNECT message generated by a successful AT&T In-Band Transfer Connect operation.

Note:

For information on NCR administration and other administration measures that are required when the AT&T In-Band Transfer Connect℠ service is used, see *Administering Avaya Aura® Call Center Elite*.

AT&T In-Band Transfer Connect℠ operations can be initiated by Call Vectoring after performing the following administration steps:

1. Administering a **`route-to number`** vector step with an announcement extension, where the associated announcement is recorded with Dial Tone Multi-Frequency (DTMF) tones that include a *T followed by a PSTN endpoint number.
2. Administering a BSR location **VDN Interflow** field on the BSR Application Plan screen as an announcement extension, where the associated announcement is recorded with DTMF tones that include a *T followed by a PSTN endpoint number.
3. Administering a BSR location **VDN Interflow** field on the BSR Application Plan screen as a local switch VDN number associated with a vector that contains an announcement step, where the associated announcement is recorded with DTMF tones that include a *T followed by a PSTN endpoint number.

When the **`route-to number`** vector in action 1 is executed or when a **`queue-to best`** vector step is executed and the BSR location described in action 2 or action 3 is selected as the BSR best location for call interflow, the AT&T In-Band Transfer Connect℠ operation succeeds, but no UUI IE information is sent to the redirected-to PSTN endpoint by Communication Manager. However, for step 3, NCR can be administered for use with the AT&T In-Band Transfer Connect℠ service and a **`disconnect hearing announcement none`** vector added after the **`announcement`** step such that UUI information associated with the call is passed to the routed-to endpoint when the call redirection is completed. This UUI information can be used in agent screen pop-ups at the redirected-to PSTN endpoint where the call is interflowed.

BSR call flow resulting in AT&T In-Band Transfer Connect UUI IE

A typical BSR call flow that results in the UUI IE being inserted in the ISDN DISCONNECT message during a successful AT&T In-Band Transfer Connect operation is as follows:

1. A PRI call from the PSTN switch arrives at the local Avaya Communication Manager and is routed to a VDN that uses a vector to do subsequent BSR vector processing.
2. The BSR polling vector steps on the local server receive status information from various local skills and remote BSR locations, and identify a remote call center site as the BSR best location.
3. Call control passes to the interflow VDN selected as the BSR best location specified on the Best Service Routing Application Plan screen.

   For information on administering a BSR application plan, see “Call Vectoring methods used with AT&T In-Band Transfer Connect service” in the *Administering Avaya Aura® Call Center Elite* document.

   Important:

   The **Net Redir** field in the BSR application plan for the remote location must be set to `n`.
4. The vector associated with the interflow VDN for the BSR best location includes the following:
   - An **`announcement`** vector step that specifies an extension for which a special sequence of DTMF digits has been recorded. The recorded DTMF digits return in-band information about the redirected-to endpoint back to the PSTN. The DTMF digits provided in the announcement are entered from a Touch-tone keypad with the following format:

     *T + PSTN number

     T corresponds to the number 8 button on a DTMF keypad and the PSTN number represents the PSTN endpoint number where the call is redirected.

     Note:

     The phone equipment required to create the announcement is described in “About setting up DTMF announcements for the AT&T In-Band Transfer Connect service” in the *Administering Avaya Aura® Call Center Elite* document.
   - A **`wait-time hearing silence`** step providing a brief interval to allow sufficient time for the PSTN switch to process the DTMF digits.
   - A **`disconnect after announcement none`** vector step. This vector step sends an ISDN DISCONNECT message that includes a UUI IE. The UUI IE contains the Avaya Information Forwarding for the call that is sent to the PSTN switch.
5. The PSTN switch makes the connection to the specified redirected-to endpoint and releases the B-channel connection with Communication Manager.

Call Vectoring methods used with the AT&T In-Band Transfer Connect service

The BSR feature invokes the AT&T In-Band Transfer Connect℠ service. You can administer an interflow VDN in the BSR Application table to route to a near-end Communication Manager VDN. The VDN causes the execution of a *T announcement vector step rather than setting the same interflow VDN number to route directly to a VDN on a far-end Communication Manager.

You must administer the BSR location on the BSR Application Plan screen to enable a BSR polling vector to identify a BSR best location and route the incoming call to the location. The BSR Application Plan must meet the following requirements:

- The plan must include at least one interflow VDN that is associated with the vectors that include the vectors steps necessary for successful invocation of the AT&T In-Band Transfer Connect℠ service.
- The **Net Redir** field associated with a location where AT&T In-Band Transfer Connect℠ is used must be set to `n`.

Example: The following example shows how you can use BSR with the AT&T In-Band Transfer Connect℠ service to implement call redirection. In the example, local interflow and not remote interflow VDN numbers are assigned on the BSR Application Plan screen.

```
                  BEST SERVICE ROUTING APPLICATION

Number: 1    Name:        Maximum Suppression Time: 60      Lock? y

Num  Location Name    Switch Node      Status Poll VDN    Interflow VDN     Net Redir?
1    Omaha              320            95022011           4004                  n
2    Paris              320            95022111           4005                  n
3    Sydney             320            95032211           4006                  n
```

The example application plan lists VDN extension numbers that are local to Communication Manager. Each VDN is associated with vectors that are designed to invoke the AT&T In-Band Transfer Connect℠ service operations.

Each vector associated with the interflow VDNs listed in the application plan includes the following steps:

```
1. announcement 1234 [*8 and PSTN number for remote site]
2. wait-time 2 secs hearing silence
3. disconnect after announcement none
```

In the vector example, step 1 provides the extension for an announcement that plays the DTMF digits. Step 2 includes a **`wait-time`** step to give the PSTN switch time to process the in-band information sent by the announcement in the preceding step before the call is disconnected at Step 3. The **`disconnect`** command sends an ISDN DISCONNECT message that includes the Information Forwarding data for the call in a codeset 0 or 7 UUI IE element.

Important:

The type of Information Forwarding data sent to the PSTN depends on how the **UUI IE Treatment** field on the Trunk Features page of the Trunk Group screen is administered:

If the **UUI IE Treatment** field is set to `Service Provider`, the ASAI user data is forwarded to the PSTN in the ISDN DISCONNECT message. If the **UUI IE Treatment** field is set to `Shared`, the call center-related data described in NCR and Information Forwarding is forwarded to the PSTN in the ISDN DISCONNECT message.

You can use the call vector example to begin the AT&T In-Band Transfer Connect℠ service with non BSR vector programming. To execute the example vector in a non-BSR vector call flow, apply the guidelines and notes related to the *T announcement format.

NCR implementation methods

NCR operations by Call Vectoring method

Note:

The following description does not apply when the AT&T In-Band Transfer and Connect℠ service is used with NCR.

1. The PSTN switch sends an incoming ISDN call to Communication Manager, where the call enters vector processing.
2. One of the following occurs:
   - If the Communication Manager trunk group and PSTN or VPN switch are configured to use an NCT-type redirection protocol, the redirecting Communication Manager returns an ISDN CONNECT message to the PSTN switch. Any of the following vector commands can be used to return the message:
     - **`announcement`**
     - **`collect`** *x* digits
     - **`converse-on split`**
     - **`wait`** hearing *music*
     - **`wait`** hearing *(announcement extension)* then (“continue”, “music”, “ringback” or “silence”)

     Note:

     If the redirecting Communication Manager does not execute one of the listed vector steps, a CONNECT message is automatically returned to the PSTN switch.
   - If the server trunk group and PSTN or VPN switch are configured to use the NCD redirection protocol, a CONNECT message is not sent to the PSTN switch. Therefore, when the NCD protocol is applied, none of the listed vector commands are included in call vectors that implement NCR.
3. Call processing proceeds to either a **`route-to number`** ~*r* or to a BSR **`queue-to best`** vector step. Depending on which type of redirection is administered for the incoming trunk group, either NCT-type or NCD processes are initiated. In either case, a FACILITY message is sent to the public network over the D-channel associated with the incoming trunk to invoke redirection of the call.

Note:

The following items pertain to the PSTN or VPN endpoint number and to the receiving vector for the interflow location.

4. The PSTN or VPN switch indicates redirection success or failure consistent with the protocol-specific operations described in NCR options supported by PSTNs. An unsuccessful NCR attempt results in one of the following outcomes:
    - If an NCT-type protocol is used, the redirecting Communication Manager establishes a trunk-to-trunk connection.
    - If the NCD protocol is used and the Avaya DEFINITY version is earlier than load 37 of Release 10, vector processing continues to the next vector step that follows the **`queue-to best`** vector step without any best local BSR call treatment.
    - If the NCD protocol is used, the call can be redirected to the best location by means of a trunk-to-trunk connection. However, the ability of the originating server to establish such a trunk-to-trunk connection depends on the specific features of the NCD protocol in use.

Using BSR queue-to best vector step to activate NCR

NCR is especially useful for multisite call center operations in which the Best Service Routing (BSR) feature is activate as the number of PRI B-channels needed for call interflows is reduced. The **`queue-to best`** vector step can be used to interflow ISDN calls between Communication Managerover the PSTN. This method provides the best approach for balancing loads across a multisite environment and is more cost effective and accurate than pre-delivery routers.

NCR is activated by the **`queue-to best`** vector step when the BSR feature determines a BSR location is the best BSR location and that location is administered with the **Net Redir** option set to `y` on the BSR Application Table screen.

Note:

The administered **Interflow VDN** field on the Best Service Routing Application screen must be a PSTN or VPN endpoint number without a trunk/ARS/AAR access codes included. For some PSTN switch dialing plans, the long-distance access code. For example, 1 in the United States must be prefixed to the PSTN number for the call to be successfully routed by the PSTN switch.

As shown in the following example, the Best Service Routing Application Plan screen must include locations that have the **Net Redir** field set to `y`.

```
                    BEST SERVICE ROUTING APPLICATION

Number: 1    Name:        Maximum Suppression Time: 60      Lock? y
Num  Location Name   Switch Node    Status Poll VDN  Interflow VDN    Net Redir?
1    Omaha                           95552011       3035551211            y
2    Paris                           95552022       18005551234           y
3    Sydney                          95552033       18665553456           y
```

An appropriate vector is then used to identify a BSR best location and NCR is activated by the queue-to-best vector step.

```
wait 2 seconds hearing ringback
consider skill 1 pri l adjust-by 0
consider location 1 adjust-by 20
consider location 2 adjust-by 40
consider location 3 adjust-by 20
queue-to best
```

Using route-to number ~r vector step to activate NCR

This method can be used to invoke NCR when a **`route-to number`** vector step that specifies a number that begins with the **~**r character. This method can be used to invoke NCR with or without the **LAI**option set to `y` or with Attendant Call Vectoring active.

Note that the administered route-to number vector step number field must be a PSTN or VPN endpoint number without a trunk/ARS/AAR access codes included. For some PSTN or VPN switch dialing plans, the long-distance access code (for example, a “1” in the United States) must be prefixed to the number for the call to be successfully routed by the PSTN or VPN switch.

Example route-to number ~r vectors: The following examples show vectors that include **`route-to number`** commands to activate NCR, either with or without use of the Attendant vectoring feature.

```
wait 0 seconds hearing ringback
goto step 4 if skill oldest-call < 30 secs
route-to number ~r13035403001
queue-to skill 35 priority m
...
```

```
goto step 6 if time-of-day is all 17:00 to 09:00
wait 0 seconds hearing ringback
queue-to attd-group
wait 999 secs hearing music
stop
route-to number ~r13035551002
```

Using vector/VDN variables with route-to number ~r to activate NCR

The **number** field of the **`route-to number ~r`** vector command can be administered with a global vector variable A-Z or AA-ZZ instead of a PSTN endpoint number after the leading *~r* characters. The **number** field of the **`route-to number ~r`** vector command can also be administered with a VDN variable V1-V9 instead of a PSTN endpoint number after the leading *~r* characters.

An example of using the **`route-to number ~r`** vector command with a vector variable in the **number** field is shown in the following example. For this example, it is required in the Variables for Vectors screen that the following administration is done:

- Vector variable A is defined as of type collect for digit-buffer and L for local

- Vector variable T as of type tod to contain the current system clock time-of-day value

```
1.  goto step 5 if T < 0700      [if time-of-day is less than 7:00 a.m., set up NCR
                                   call-redirection to out of hours PSTN endpoint]
2.  goto step 5 if T > 1800      [if time-of-day is after 6:00 p.m., set up NCR
                                  call-redirection to out of hours PSTN endpoint]
3. set A = none CATR 18005555555   [set digit-buffer to in-office
                                                 hours PSTN endpoint number]
4. goto step 6 if unconditionally [jump to step 6 to do NCR call-redirection ]
5. set A = none CATR 18661111111 [set digit buffer to out-of-office hours
                                               PSTN endpoint number]
6. route-to number ~rA [initiate NCR call-redirection operation]
```

NCR activation using ASAI Call Transfer and third-party Merge/Release operations

NCR NCT-type operations are activated by ASAI call processing when the Call Transfer or Third-Party Merge/Release operation is performed by a CTI application. This occurs in the following manner:

1. This is typically initiated by the CTI application user selecting an icon, menu item, or button to transfer an answered incoming ISDN call to another party over the PSTN.

   Since the incoming ISDN call must be connected to a station user before the Call Transfer or Third-Party Merge/Release operation is requested, NCR can only initiate the call redirection if an NCT-type protocol is optioned on the trunk.
2. If a call arrives at an ASAI-monitored VDN, ASAI will send appropriate information in the ASAI disconnect event to notify the CTI application that the call has been redirected by NCR.

For the ASAI operations listed above to succeed, the following conditions must be in effect:

- The **Network Call Redirection** field is set on the System Parameters Customer Options screen.
- An NCR NCT-type protocol is administered for both the incoming and outgoing call ISDN trunk group.
- The PSTN number that the CTI application uses to redirect an incoming ISDN call to another PSTN endpoint must be added to the ARS digit analysis screen in such a way that for the NCR MCI NCT and TBCT protocols, the second leg of the call transfer uses the same trunk group with a trunk that has the same D-channel as the incoming call. For the NCR ETSI ECT protocol, the CTI-initiated second call leg can be over a different trunk group with a different signaling group than the incoming call leg.

Note:

NCR-related AAR/ARS routing table administration is required for station transfer or conferencing with MCI trunks.

Other things to know about using NCR with ASAI

Using ASAI data for call tracking

ASAI event reporting allows tracking of ISDN ACD calls that were redirected by NCR in a multi-server call center environment. These calls can be tracked by the UCID assigned to each call, or by the UUI information inserted by the application through either the Third Party Make Call or Adjunct Routing features.

ASAI drop event

Successful NCR call redirection causes an ASAI drop event to be sent to the CTI application with a CV_REDIR cause value of decimal (30) after the redirection is completed. Only one NCR drop event is received for a successful NCR operation when the NCT PSTN feature is used, even though two trunks are dropped by the PSTN.

ASAI third-party merge/call transfer

The CTI application requests a third-party merge/call transfer ASAI operation to transfer the call to the second Communication Manager. This is only used if Network Call Transfer is not available. Once the two calls merge, then ASAI sends a third-party acknowledgement, and when the call is completed, ASAI sends a drop event report, and the third-party call ends.

NCR activation using station call transfer or call conference/release operations

When an incoming ISDN call over a trunk with NCT-type PSTN service is answered at a station or voice response unit (VRU), the station user or VRU places the call on hold, and dials the number for a PSTN of VPN endpoint where the outgoing trunk B-channel is determined by AAR or AAS routing. The station user initiates a call transfer using the Transfer feature button or a switch hook flash, or the VRU initiates a call transfer by using an analog or line-side E1/T1 switch-hook flash.

The switch automatically sends an “invoke NCT” FACILITY message when the transfer is completed if the following conditions are met:

- The **Network Call Redirection** field is set to `y` on page 3 of the System Parameters Customer Options screen.
- An NCT-type protocol is administered for both the incoming and outgoing call ISDN trunk group.
- The second leg call is eligible for redirection by means of an NCT-type protocol, which requires for the MCI NCT and TBCT protocols the second leg of the call is in the same trunk group and has the same signaling group as the incoming call. For the NCR ETSI ECT protocol, the second leg of the call can be over a different trunk group with a different signalling group than the incoming call leg.

If the station user or IVR initiates and completes a three-way conference instead of doing a call transfer operation as above, and releases or hangs up from the conference with the following condition also being met, the switch automatically sends an “invoke NCT” ISDN message to the PSTN or VPN switch if also the following condition is met:

- The number of parties in the conference including the conference originator must be no greater than three parties.

Note:

NCR-related AAR/ARS digit-analysis and routing table administration is required for correctly setting up the second call leg over NCT-type trunks associated with the station or IVR call transfer and call conference/release operations.

NCR activation using ASAI adjunct route operations

NCR can be invoked by specifying the activate NCR option for the ASAI route message in a route select ASAI message sent by a CTI application to the Avaya Communication Manager after an adjunct routing vector step is executed during call vector processing. This Communication Manager capability provides greater flexibility for CTI applications to directly route calls to PSTN or VPN endpoints without the need to specify a VDN extension in the route select ASAI message to route the call instead to a VDN and vector step that activates NCR via a route-to number ~r or queue-to-best vector step. The invocation of NCR by the adjunct routing vector step route select ASAI message for various NCT-type protocols follows the same rules as used for the route-to number ~r or queue-to-best vector step operations.

For more information, see the following ASAI documents:

- For information on the call options codepoint for NCR routing or the ASAI Call Route Selection message, see the *ASAI Protocol Reference* document.
- For information on feature interactions, see the *ASAI Technical Reference* document.

NCR considerations

Limitations of call redirection

Limitation	Description
NCR feature support	PSTN support for NCR varies with the geographical location and can be limited or absent in some areas. Consult your Avaya account team to determine the PSTN service provider availability of the NCR protocols in your area.
NCD redirection protocol support	At this time, no PSTNs offer the Network Call Deflection “retain call until alerting or connect” operation. Therefore, only the Network Call Deflection “clear call upon invocation” offer is available from PSTNs. Both methods are described in this document. Negotiate with your PSTN as the NCR feature works on either platform. NCR is limited by which PSTN platform is available to you.
Allowable number of redirection per call	There can be limits placed on the number of times a call can be redirected over the public network. The limits are imposed by the public network service provider. For example, in the United States, MCI currently allows only one redirection per call. In the United Kingdom, there is a limit of 20 call deflections per call. In addition, there can be additional charges associated with redirected calls.

Table continues…

Limitation	Description
User-to-User information forwarding support	Some public network service providers do not support forwarding of UUI, including ASAI user data, collected digits, VDN name, the VDN in-time, as reflected by the NETINTIME database items, and the UCID. In such situations, Information Forwarding is lost and the second leg of the redirected call looks like an entirely new call to the redirected-to server at the second location. One of the data items lost is the VDN name, which is rerouted to the originally called service DNIS information. Call forwarding indication can be achieved by using dedicated VDNs for call forwarding, but this strategy loses the benefits of Information Forwarding inherent with NCR and limits use of CTI applications. PSTN service providers typically charge by call or by a monthly rate for the redirect and UUI transport services. For more information on charges, contact your Avaya account team.

NCR operational considerations

Reserving outbound trunk B-channels to ensure NCR operations succeed

When the trunk group service type is set to `call-by-call`, the trunk group Usage Allocation capability can be used to reserve a minimum number of trunk channels for outgoing PRI B-channel calls within the same trunk group and same D-channel.

For more information, see "Reserving trunk group B-channels for NCT-type redirection operations" in the *Administering Avaya Aura® Call Center Elite* document.

Call Vectoring configuration required for successful MCI NCT operations

When NCR is used with the MCI NCT protocol, the VDN call vector the call is redirected to by a successful MCI NCT operation must immediately return an ISDN CONNECT message to the PSTN switch. To meet this requirement, either a **`wait`** *0 secs* hearing *music* or an **`announcement`** vector step must be the first step executed in the redirected-to call vector.

Configuring Ericsson AXE-10 for ETSI ECT operations

Before you begin

Verify that AXE-10 contains VN7 Translocal 4.2 or later software. This software is also called GOAS 2.1 in Ericsson.

Procedure

1. Configure AXE-10 for the `pure ETSI` level.
2. To subscribe to the AXE-10 `ETSI ECT` mode, configure all the PRI trunks used with Communication Manager NCR, or the ETSI ECT feature.
3. On the AXE-10 Trunk Configuration screen, enter the **ECT** category as `ON`.

   Note:

   Do not configure the AXE-10 PRI trunk to expect a "HOLD" ISDN message when the NCR ETSI ECT feature sends a message as a part of the ETSI ECT invocation sequence.

NCR and Information Forwarding

The Avaya Information Forwarding feature is supported with NCR when the PSTN supports ISDN UUI IE transport in conjunction with the specific network redirection protocol used by the switch.

UUI data included in Information Forwarding for ISDN calls

Information Forwarding forwards the following call center-related data (as User-to-User Information) with an ISDN call:

- Adjunct Switch Application Interface (ASAI) user data
- Universal Call ID (UCID)
- Collected digits
- In-VDN time
- VDN name

UUI data forwarding

When an NCT-type option is used for NCR, the UUI is forwarded by the Avaya Communication Manager in the ISDN SETUP message sent with the call to the second site.

When the NCD option is used for NCR, the UUI is included in the ISDN FACILITY invoke message sent from the Avaya Communication Manager to the PSTN. The PSTN then forwards the UUI to the second site.

When the AT&T In-Band Transfer and Connect service is used for NCR, the UUI is returned in an ISDN DISCONNECT message that includes the data in a codeset 0 or 7 UUI IE element.

PSTN terms used for UUI transport service

For NCT-type options and the NCD option, the PSTN service provider must configure the PRI trunks used with the NCR feature to transport the UUI data associated with the Information Forwarding feature.

The various PSTN terms used in different countries for UUI IE transport are listed in the following table.

Country	UUI transport term	Providers
Australia	UUS Service 1	Telstra
Canada	UUS Service 1	AT&T/Canada Bell Canada
France	Mini-Message	France Telecom
Germany	(included in basic ISDN package)	Deutsche Telecom
Singapore	Not supported	
UK	Not supported	
USA	N-Quest Type 1 Service MA UUI Type 1 Service	Verizon AT&T

NCR interactions

- Attendant Vectoring: Can use the **`route-to number`** vector step to route calls to attendants located at another Communication Manager node. The operation of the NCR feature using the NCT-type or NCD networks features to accomplish the call redirection is exactly the same as for redirecting ACD calls.
- Advice of Charge: No new capabilities are added for the NCR feature for the Advice of Charge PSTN feature. Use the Advice of Charge feature with the same trunk facilities used for the NCR feature.
- BCMS: No change is made to BCMS for support of NCR. Redirected calls are tracked as completed calls since the PSTN disconnects the incoming facility of the original call when the call is redirected to another site.
- Enhanced Information Forwarding: For the NCR feature, Enhanced Information Forwarding transports UUI for the incoming ISDN call to the PSTN endpoint that receives the redirected call. The use of the Enhanced Information Forwarding capability with NCR requires that the incoming call trunk group be assigned as `shared`, that is, the **UUI IE treatment** field is set to `shared`. However, if the trunk group is set up as `service provider`, only the ASAI user information or user information provided by the incoming ISDN call, is included in the UUI IE sent on a non-shared basis to the redirected-to PSTN endpoint. NCR supports Information Forwarding for AT&T In-Band Transfer and Connect service.
- Look-Ahead Interflow: NCR activation using the **`route-to number`** vector step does not require LAI to be active to provide multisite capabilities, which are required to check remote locations and to access the BSR Application Plan screen.
- Service Observing by VDN: If the Service Observing by VDN feature is used to service observe a VDN, where the NCR feature is used to redirect incoming ISDN calls, the service observer hears the same tones, music, or announcements heard by the incoming caller before the NCR feature reroutes the call to another PSTN endpoint. When the NCR operation is completed, the service observer is dropped as an observer of the incoming call and placed in the service observing queue associated with the VDN.
- Trunk-to-Trunk Transfer: If you enable the NCR feature and use the ASAI Third-Party make Call/Transfer operation to redirect an incoming ISDN to a PSTN endpoint, set the **Trunk-to-Trunk Transfer** field on the System-Related Customer-Options screen to `y` for the call redirection to succeed. If the route-to number or BSR queue-to best vector step uses the NCR feature to redirect an incoming ISDN call to a PSTN endpoint, the **Trunk-to-Trunk Transfer customer** option does not need to be set to `y`.

  For more information, see “Using the route-to command for NCR” in the *Programming Call Vectoring Features in Avaya Aura® Call Center Elite* document.
- VDN Return Destination: If you administer the VDN Return Destination (VRD) feature for the VDN associated with a vector that causes the NCR feature to be invoked, VRD is canceled when the call is redirected by NCR.
- CMS Database Items: The following Avaya CMS database items are affected by NCR:
  - DEFLECTCALLS: In the VDN CMS database tables, the DEFLECTCALLS item includes the number of calls that are redirected using NCR through the BSR feature by using

the `route-to number` or `queue-to best` commands. Successful NCR attempts are pegged as DEFLECTCALLS.

- INTERFLOWCALLS: In the VDN CMS database tables, the INTERFLOWCALLS item includes successful BSR interflows using NCR redirections.
- LOOKATTEMPTS: In the VDN CMS database tables, the LOOKATTEMPTS item includes the number of times the LAI or BSR interflow was attempted for calls in the vector. Successful LAI or BSR attempts are also counted. NCR invoke attempts (NCD or NCT) are also reflected in LOOKFLOWCALLS.
- LOOKFLOWCALLS: In the VDN CMS database tables, the LOOKFLOWCALLS item includes the number of INTERFLOWCALLS redirected by the LAI or BSR features. LOOKFLOWCALLS is a subset of INTERFLOWCALLS and includes LOOKATTEMPTS for the LAI or BSR interflows. With BSR interflow using trunk-to-trunk transfer or NCR, every LOOKATTEMPT is also be counted as a LOOKFLOWCALLS unless a failure occurs.

NCR support with SIP

Network Call Redirection (NCR) provides Communication Manager call routing method between sites on a public network or a Virtual Private Network (VPN) that can reduce trunking costs. The cost reductions are particularly valuable in enterprises or multisite call center environments where trunk costs are high.

When an incoming call arrives at a Communication Manager server that has the NCR feature activated, call redirection is managed by the SIP service provider or VPN switch instead of the local Avaya server. As a result, trunks that the server otherwise retains to accomplish a trunk-to-trunk transfer are released after the call redirection takes place.

The cost reductions associated with reduced trunk use can be significant particularly when Avaya virtual routing features such as BSR with Expected Wait Time (EWT), are implemented. The cost-savings are achieved by the Avaya customer requiring fewer trunks to handle the same number of incoming or outgoing calls after the NCR feature is implemented within the local Communication Manager.

SIP Network Call Redirection protocols

NCR can occur over SIP trunks. The SIP REFER or SIP 302 Moved Temporarily messages contain all the information needed for Network Call Redirection. When the call is answered by an agent or call vectoring event, a SIP REFER message is sent. If a SIP call is not answered by an agent or a call vector process, a SIP 302 Moved Temporarily message is sent.

SIP limitations on call redirection

NCR feature support

SIP service provider support for NCR varies with the geographical location and can be limited or absent in some areas. Consult your Avaya account team to determine the SIP service provider availability of the NCR protocols in your area.

Allowable number of redirection per call

There can be limits placed on the number of times a call can be redirected over the public network. The limits are imposed by the public network service provider. There can be additional charges associated with redirected calls.

User-to-User information forwarding support

Some public network service providers do not support forwarding of UUI, including ASAI user data, collected digits, VDN name, the VDN in-time, as reflected by the NETINTIME database items, and the UCID. In such situations, Information Forwarding is lost and the second leg of the redirected call looks like an entirely new call to the redirected-to server at the second location.

One of the data items lost is the VDN name, which is rerouted to the originally called service DNIS information. The indication that the call has been forwarded can be achieved by using dedicated VDNs for call forwarding, but this strategy loses the benefits of Information Forwarding inherent with NCR and potentially limits use of the CTI applications.

SIP NCR and Information Forwarding

Information Forwarding is supported with NCR when the SIP service provider supports UUI transport in conjunction with the specific network redirection protocol used by the switch.

UUI data included in Information Forwarding for a SIP call

Information Forwarding forwards the following call center-related data (as User-to-User Information) with a SIP call:

- Adjunct Switch Application Interface (ASAI) user data
- Universal Call ID (UCID)
- Collected digits
- In-VDN time
- VDN name

UUI data forwarding with SIP

When NCR is used, the UUI is forwarded by the Avaya Communication Manager in the SIP REFER or 302 Moved Temporarily messages.

The UUI is forwarded by the Avaya Communication Manager in a SIP REFER message if the call has been answered. A call can be answered for example, if an agent answers the call, a vector plays an announcement or music, or the call is processed by a command that provides an answer such as collect digits. The REFER is sent back to the caller, causing the first call to be dropped after a second call is issued and established with the next location.

The UUI is forwarded by the Avaya Communication Manager in a 302 Temporarily Moved message if NCR is invoked before the call is answered. The first call is redirected to the next location.

NCR operations for SIP by Call Vectoring method

The processes by which NCR is implemented for SIP by a Call Vectoring method are summarized in the following steps:

1. The SIP service provider switch sends an incoming call to Communication Manager, where the call enters vector processing.
2. Call processing proceeds to either a **`route-to number`** *~r* or to a BSR **`queue-to best`** vector step.

BSR queue-to best vector step activation of NCR

You can use NCR to reduce the number of trunks for call interflow. This method provides the best approach for balancing loads across a multisite environment and is more cost effective and accurate than pre-delivery routers. You can use the **`queue-to best`** vector step to interflow calls between Communication Manager over the SIP service provider network. You can use SIP to interflow BSR calls, but not BSR polling. You can use BSR polling through other methods such as Polling Over IP without B-Channel.

The **`queue-to best`** vector step activates NCR when the BSR feature determines that a particular BSR location is the best location and that location is administered with the **Net Redir** field set to `y` on the BSR Application Table screen. Note that the administered **Interflow VDN** field on the Best Service Routing Application screen must be a SIP service provider or VPN endpoint number without a trunk/ARS/AAR access codes included. For some SIP service provider switch dialing plans, you must prefix the long-distance access code, for example, 1 in the United States, to the SIP service provider number for the call to be successfully routed by the SIP service provider switch.

As shown in the following example, the Best Service Routing Application Plan screen must include locations that have the **Net Redir** field set to `y`.

```
                  BEST SERVICE ROUTING APPLICATION

Number: 1    Name:        Maximum Suppression Time: 60      Lock? y

Num  Location Name   Switch Node     Status Poll VDN  Interflow VDN    Net Redir?
1    Omaha                            95552011       3035551211             y
2    Paris                            95552022       18005551234            y
3    Sydney                           95552033       18665553456            y
```

An appropriate vector is then used to identify a BSR best location and NCR is activated by the **`queue-to-best`** vector step.

```
wait 2 seconds hearing ringback
consider skill 1 pri l adjust-by 0
consider location 1 adjust-by 20
consider location 2 adjust-by 40
consider location 3 adjust-by 20
queue-to best
```

SIP NCR interactions

- Attendant Vectoring: Use the **`route-to number`** vector step to route calls to attendants located at another Communication Manager node. The operation of the NCR feature using the network call redirection features to accomplish the call redirection is exactly the same as for redirecting ACD calls.
- Basic Call Management System (BCMS): No change is made to BCMS for support of NCR. Redirected calls are tracked as completed calls since the SIP service provider disconnects the incoming facility of the original call when the call is redirected to another site.
- Enhanced Information Forwarding: For the NCR feature, Enhanced Information Forwarding transports User-to-User information (UUI) for the incoming call to the SIP service provider endpoint that receives the redirected call. The use of the Enhanced Information Forwarding capability with NCR (the recommended configuration) requires that the incoming call trunk group be assigned as shared, that is, the **UUI treatment** field is set to `shared`. However, if the trunk group is set up as service provider, only the ASAI user information (or user information provided by the incoming call) will be included in the UUI sent on a non-shared basis to the redirected-to SIP service provider endpoint.
- Look-Ahead Interflow (LAI): NCR activation using the route-to number vector step does not require LAI to be active to provide multisite capabilities, which are required to check remote locations and to access the BSR Application Plan screen.
- Service Observing by VDN: If the Service Observing by VDN feature is used to service observe a VDN, where the NCR feature is used to redirect incoming calls, the service-observer will hear the same tones, music, and/or announcements heard by the incoming caller before the NCR feature reroutes the call to another SIP service provider endpoint. When the NCR operation is completed, the service-observer will be dropped as an observer of the incoming call and placed in the service-observing queue associated with the VDN.
- Trunk-to-Trunk Transfer: If the NCR feature is optioned and the ASAI Third-Party make Call/transfer operation is used to redirect an incoming call to a SIP service provider endpoint, the **Trunk-to-Trunk Transfer** field on the System-Parameter Features screen must be enabled for the call redirection to succeed. If the route-to number or BSR queue-to best vector step uses the NCR feature to redirect an incoming call to a SIP service provider endpoint, the Trunk-to-Trunk Transfer customer option does not need to be set to `y`.
- VDN Return Destination (VRD): If VRD is administered for the VDN that is associated with a vector that causes the NCR feature to be invoked, VRD is canceled when the call is redirected by NCR.
- CMS database items: The following database items are affected by NCR:
  - DEFLECTCALLS: In the VDN CMS database tables, the DEFLECTCALLS item includes the number of calls that are redirected using NCR through the BSR feature by using the route-to number or queue-to best commands. Successful NCR attempts are pegged as DEFLECTCALLS.
  - INTERFLOWCALLS: In the VDN CMS database tables, the INTERFLOWCALLS item includes successful BSR interflows using NCR redirections.

- LOOKATTEMPTS: In the VDN CMS database tables, the LOOKATTEMPTS item includes the number of times the LAI or BSR interflow was attempted for calls in the vector. Successful LAI or BSR attempts are also counted. NCR invoke attempts are also reflected in LOOKFLOWCALLS.
- LOOKFLOWCALLS: In the VDN CMS database tables, the LOOKFLOWCALLS item includes the number of INTERFLOWCALLS that were redirected by the Look-Ahead Interflow or BSR features. LOOKFLOWCALLS is a subset of INTERFLOWCALLS and includes LOOKATTEMPTS for the Look-Ahead Interflow or BSR interflows. With BSR interflow using trunk-to-trunk transfer or NCR, every LOOKATTEMPT will also be counted as a LOOKFLOWCALLS unless a failure occurs.

Percentage Allocation Routing

Use Percentage Allocation Routing to distribute calls among a set of call centers or VDNs based on a specified percent allocation. Communication Manager can initially direct incoming calls that arrive at a particular VDN to a Policy Routing Table (PRT) for Percent Allocation instead of to a vector. The PRT distributes the calls to the administered Route-to VDNs based on the administered percent allocation targets.

This feature is useful for segmented call-handling, outsourcing, and optimizing call handling in a multiple-location enterprise. With Percentage Allocation Routing, you can allocate target percentages and, for example, do the following:

- Allocate certain call types among multiple answering groups with similar skills.
- Allocate maximum calls to a more economical calling group.
- Ensure that the organization meets the terms of a service level agreement.

Considerations for implementing percentage allocation routing

- Modification to the table while calls are being routed to the VDN(s) that the PRT is assigned to can result in calls being miss-routed, miss-appropriated or other indeterminate actions. To prevent such a situation from occurring, create a new PRT table with the required changes and replace the old version.
- The following are the valid entries in the **Period** field:
    - `100_count` (default): The call count, displayed in percentage, is reset after the total calls for the PRT reach 100 which is when the total calls match the target routing pattern percentages. This ensures that the routing points have equal distribution of calls all the time.
    - `max_count`: The call count is maintained until calls delivered to one of the VDNs exceed 65,400. At that point, calls are continued to be distributed over the VDNs, but the call counts are reset when the actual percentages equal the targets for all of the VDNs at the same time.

- `Half-hour`: Resets the call count at the top of the hour and at the 30 minute point.
- `Hour`: Resets the call count at the top of the hour.
- `Daily`: Resets the call count at midnight, every night.
- `Weekly`: Resets the call count at midnight on Saturday.

- The actual percentage and the call count are reset whenever the PRT screen is changed.
- Calls are routed to the VDN destination that is farthest from meeting its target allocation.
- Calls are routed based on the actual call count, not on the actual percentage. When there is no actual call count, at startup or after reset, the route point with the highest target is selected.
- Calls routed through a PRT are reported to BCMS, CMS, and Avaya IQ as though the calls were routed to a VDN assigned to vector 0. VDN reports can be created using the existing CMS custom reporting capabilities to show the percentages and the number of calls that have been distributed among the destination VDNs. Avaya IQ provides additional reporting including what VDN the call was routed to, the PRT table number and the type used, and the matched percentage.
- A PRT can be assigned to multiple VDNs. There is no restriction on the VDN that can be entered as a route point.
- There are limits for the number of PRT tables that can be defined and there is a system routing point (PRTs x VDN destinations) limit. Both the limits can be found in the Communication Manager System Capacity tables and displayed on the Display Capacity screen.
- In addition to add, change, display, list, and remove, the following commands also support PRT:
  - `list vdn`: Displays destination type, Vector or PRT.
  - `list usage policy-routing-table`: Lists the VDNs that use the specified PRT.
  - `list history`: Displays the history of add and change policy-routing-table commands.
  - `list trace vdn`: Displays calls to a PRT.
- When the destination is a PRT, the **Attendant Vectoring** and **Meet-Me Conferencing** fields do not appear.
- Routing through the PRT to the selected VDN is considered as a **`route to`** command operation and this triggers the *Allow VDN Override* rule that is assigned the previous VDN.
- For VDN domain monitoring with CTI/ASAI, a call to a VDN, that is, routed via a PRT. appears as if the call was routed to the destination VDN by a **`route-to number`** command in a vector assigned to the original VDN.
- To evenly distribute calls across three routing points, administer the routing points with a 33 percent target, add a fourth routing point that routes back to the PRT with a target of 1 percent. Set the **Period** field to `100_count` and calls are distributed evenly.

Rules for percentage allocation routing

- The target percentages must be in integers. You cannot use decimals or fractions.
- The target percentage of all the VDNs must add up to 100 percent before form submittal.

Percentage allocation routing example

Use the **Destination** and **Number** fields on the Vector Directory Number screen to specify the routing destination as either a vector or a PRT table. To implement percentage allocation routing, specify the routing destination as a PRT table (Destination: Policy Routing Table and Number: PRT table number).

Use the Policy Routing Table screen to specify various routing destinations and percentages allocated for each destination. The following example Policy Routing Table screen includes the Target and Actual traffic routed to each of the VDNs.

```
 display policy-routing-table 1957

                              POLICY ROUTING TABLE
 Number: 1957  Name: % distribution      Type: percentage   Period: max count


                                                 Target Actual Call
 Index Route-to VDN  VDN NAME                     %      %      Counts
   1   2220071       Gizmo support                25     27.2      3
   2   2221501       Ultra support                 5      9.0      1
   3   2220601       Customer Service South       35     27.2*     3
   4   2220511       Outsourcer Charlie           10      9.0      1
   5   2220501       Survey after service         10      9.0      1
   6   2220072       Outsourcer International     15     18.1      2
   7
   8
   9
  10
  11
  12
                                          Totals 100             11

Command:
```

Service Hours Table Routing

Service Hours Table Routing simplifies the vectors for handling calls based on office hours. Vectors use the service hours routing tables to determine how to handle calls that are received during working and non working hours. Customers can use this feature as an alternative to Time of Day (TOD) routing and can specify the working hours on a daily or hourly basis. You can administer as many as 999 different tables and use the tables to make vectoring decisions. One simple vector command can check if the call meets the administered service hours.

Without Service Hours Table Routing, customers have to add multiple TOD steps in the vectors to define the hours of operation for a specific business application.

Service Hours Table Routing considerations

- Service Hours Table Routing is not available when upgrading from a previous release.
- **Vectoring (Basic)** must be enabled.
- The Call Center Release field must be set to 4.0 or later.

Goto processing for Service Hours Table Routing

When vector processing encounters a goto if service-hours step, it determines if the current day of week and time is within the service hours listed in the corresponding table. This information is used to decide if the `goto` condition is true or false, and therefore, whether or not to go to the given step or vector. The day of week and time match is based on the system time on the Communication Manager that receives the call. The time used in the calculations is the time the call reaches the goto step.

Time adjustments on the Service Hours Table screen

The time used in the calculations can be adjusted using the **Use time adjustments from location** field on the Service Hours Table screen. This field indicates the location number on the Locations screen that specifies how the adjustments are performed.

You can make the following time adjustments using the **Use time adjustments from location** field:

- Adjust the daylight saving time from the system time
- Apply the time zone for a specific location
- Apply the daylight saving time for a specific location

If this field is blank, no adjustments are made.

Service Hours Table

Procedure

1. At the command prompt, type `change service-hours-table xxx`, where *xxx* is number of a service hours table. Press **Enter**.
2. Administer the following fields on the Service Hours Table screen:
   - **Description**
   - **Number**
   - **Start and End**
   - **Use time adjustments from location**
3. Press **Enter** to save the changes.

Goto step command for Service Hours Table Routing

Syntax 1

goto step x if service-hours in table y

The command directs the call to a specific vector step if the conditions of the call match the service hours specified in the Service Hours Table.

Syntax 2

goto step x if service-hours not-in table y

The command directs the call to a specific vector step if the conditions of the call do not match any of the service hours that are in the specified Service Hours Table.

Goto vector command for Service Hours Table Routing

Syntax 1

goto vector x @step z if service-hours in table y

The command directs a call to a specific vector if the conditions of the call match service hours that are in the specified Service Hours Table.

Syntax 2

goto vector x @step z if service-hours not-in table y

The command directs the call to a specific vector if the conditions of the call do not match any of the service hours that are in the specified Service Hours Table.

Service Hours Table Routing scenario

The following is a very basic scenario without time adjustments:

Basic Service Hours Table scenario

```
                              SERVICE HOURS TABLE

                                Number: 99
                      Description: Call-ahead Reservations_____
              Use time adjustments from location: 2__

      MON           TUE           WED           THU           FRI
  Start  End    Start  End    Start  End    Start  End    Start  End
  08:00 12:30   10:30 02:00   __:__ __:__   __:__ __:__   __:__ __:__
  13:00 16:30   15:00 20:30   __:__ __:__   __:__ __:__   __:__ __:__
  __:__ __:__   __:__ __:__   __:__ __:__   __:__ __:__   __:__ __:__
  __:__ __:__   __:__ __:__   __:__ __:__   __:__ __:__   __:__ __:__
  __:__ __:__   __:__ __:__   __:__ __:__   __:__ __:__   __:__ __:__

                           SAT           SUN
                       Start  End    Start  End
                       __:__ __:__   __:__ __:__
                       __:__ __:__   __:__ __:__
                       __:__ __:__   __:__ __:__
                       __:__ __:__   __:__ __:__
                       __:__ __:__   __:__ __:__
```

VECTOR 1:

goto vector 2 @step 1 if service-hours not-in table 99

<Service hours – Call-ahead Reservation processing>

VECTOR 2:

<After hours processing>

The following table shows how calls at different times will be processed.

Day of week or time of call	Which processing
Sunday or any time	After hours
Tuesday or 14:59	After hours
Tuesday or 15:00	Service-hours
Monday or 12:30	Service-hours
Monday or 12:31	After hours

The same scenario with time adjustments:

```
                                DAYLIGHT SAVING RULES
Rule          Change Day                 Month    Date    Time    Increment
 0: No Daylight Saving
 1: Start: first Sunday       on or after April     1    at 02:00   01:00
     Stop: first Sunday       on or after October   25   at 02:00
 2: Start: first Sunday       on or after April     1    at 02:00   02:00
     Stop: first Sunday       on or after October   25   at 02:00
```

```
LOCATIONS
ARS Prefix 1 Required For 10-Digit NANP Calls? y
 Loc.    Name          Timezone  Rule    NPA                            Proxy Sel.
 No.                   Offset                                           Rte. Pat.
 1:     Main          + 00:00     1
 2:     Branch        + 02:00     2
 3: :
 4: :
 5: :
 6: :
 7: :
 8: :
 9: :
10: :
11: :
12: :
13: :
14: :
```

The following table shows how calls at different times will be processed.

System time	Daylight saving	Adjustments	Adjusted time	Which processing
Tuesday or 07:30	yes	- subtract 1 hour (put system into standard time) - add 2 hours (location time zone) - add 2 hours (location DST)	Tuesday or 10:30	Service-hours
Tuesday or 11:01	yes	- subtract 1 hour (put system into standard time) - add 2 hours (location time zone) - add 2 hours (location DST)	Tuesday or 2:01	After hours
Monday or 06:00	no	- add 2 hours (location time zone)	Monday or 08:00	Service-hours
Monday or 10:31	no	- add 2 hours (location	Monday or 12:31	After hours

Variables in Vectors

Use Variables in Vector (VIV) to achieve the following objectives:

- Improve the efficiency of vector administration
- Provide increased manager and application control over call treatment
- Create vectors to meet the needs of the callers and the call center operations

You can define vector variables in a central variable administration table and change the assigned values by means of special vectors, VDNs or Feature Access Codes (FACs).

Based on the variable type, variables can use either call-specific data or fixed values that are identical for all calls. In either case, you can reuse an administered variable in many vectors.

Variable parameters

VIV enhances Call Vectoring by including letters from A to Z and AA to ZZ in many vector commands as conditionals and thresholds.

When you define vector variables in the centralized administration table, ensure that you assign an alphabetical designation to each variable. The designation can range from A to Z or from AA to ZZ . You can define up to 702 variables. Each variable can have only one definition. Once defined, the variables have the same type and assignment characteristics for every vector in which the variables are used. Depending on the variable type, specify some or all of the following parameters when you create a new vector variable.

Parameter	Description
Variable type	VIV provides a number of different variable types that you can use for different purposes. The kind of information associated with a variable can be call related, such as active vdn for the call, asai user information, last agent login identifier, or the time of day when a call arrives. You can use other types of variables to assign values and then use the as signals for high-level control over call processing operations. For example, you can use a single-digit value variable to test for operational states specific to your call center operations.
Scope	The scope of a variable indicates how variable values are assigned and used in vectors in which the variable appears. Variable scope can be local, local persistent for collect variable only, or global. Local variables use data associated with a call and the value assigned to the variable apply only within the original vector processing for the call. The value is cleared after the call leaves vector processing. Local persistent variables use data associated with a call and apply the data to more than one vector that processes the call. The last assigned value is retained throughout the life of the call. Global variables are system-wide and apply to all vectors in which the variables are used.
Length	Some variables require that you specify a string length that is applied when a value is assigned to the variable. In most cases, the string length actually represents a maximum bound, since most variables can use a value that has a shorter string length than that which is specified.
Start Position	If you create a variable that requires a string length, you must also specify a start position. The start position indicates the beginning position of the digit string to be assigned to the variable. The start position and the string length allow assigning only a portion of the data available to the variable.
Assignment	If you use a variable that has a user-defined value, provide the value in the **Assignment** field of the variables administration table.
Variable Access Code (VAC)	When you define a value variable, you can also set up an associated FAC. You can then dial the FAC to reset the variable assignment.

Implementing vector variables

Procedure

1. Define the variable application. Determine how to use the new variable and identify the defining characteristics of the new variable.

   Use the information to identify a variable type that meets the call center needs. For an overview of variable types and purposes, see “Variables in Vectors job aid”.
2. From the system administration terminal, enter the **`change variables`** command to bring up the Variable for Vectors administration table.
3. In the Variable for Vectors administration table, select an unused variable name between A - Z or AA - ZZ. The variable name is used to represent the variable in vector steps. In

the table row for the variable that you have selected, enter the following information in the specified fields:

a. **Description** - Enter a short description.
b. **Type**- Select the variable type.
c. **Scope** - Local, local persistent for collect variable only, or global.
d. **Length** - Enter length of the digit string.
e. **Start** - Enter digit start position, first position is 1.
f. **Assignment**- Enter an initial value.
g. **VAC** - Optional for value variables only.

Note:

Based on the variable type you select, some fields can be predefined or not applicable.

4. Perform the following steps to administer a value variable type in the **VAC** field and to use a dial procedure within the local Communication Manager to change the variable assignment using an FAC:
   a. From the system administration terminal, enter the `change feature-access-codes` command. Go to page 7 of the Feature Access Code screen.
   b. Select an unused FAC and note the associated Vector Variable Code (VVx). Possible VVx values range from VV1 to VV9.
   c. In the **Code** field, enter the digits to be dialed when you access the FAC.
   d. Go back to the Variables for Vectors administration table and enter the VVx number in the VAC column for the value variable that you are administering.
5. Program more than one vector with the selected variable using **`goto`** steps and other vector commands such as **`route-to number`**.

   You must conform to the vector syntax rules specified in command syntax for vector variables.
6. You can change the variable assignments.

   Some variables such as the *ani* and *tod* variable types, do not require value reassignments after the variables are implemented in vectors, since values for the variable are always provided by individual callers or Communication Manager. However, with other variable types, you can change the variable assignment even as calls are being processed. For example, if you use a *collect* variable in a vector step, the variable value changes when an announcement prompts a caller to enter new digits or when you use a **`set`** command.

   Note:

   When *collect* variables are provided specifically for supervisor or manager use, the *collect* variable has a global scope and the variable is applied in a special vector intended for the supervisor or manager.

Command syntax for vector variables

Announcement command

You can enter a vector variable between *A-Z* or *AA-ZZ* as an announcement extension in all commands that use an announcement in the extension field.

The following syntax rules apply when you use vector variables with the **announcement** command:

```
announcement [A-Z, AA-ZZ]
collect [1-16] digits after announcement [A-Z, AA-ZZ] for [A-Z, AA-ZZ]
disconnect after announcement [A-Z, AA-ZZ]wait-time
[0-999] sec hearing [A-Z, AA-ZZ] then
[music, ringback, silence, continue]
```

Requirements and considerations

The requirements for using variables in place of specific entries in vector commands are follows:

- You can use a VDN or a vector variable, but not both.
- When the command is executed, the assignment entry for that variable is based on the type assignment administered in the Variables for Vectors table or VDN screen for the active VDN for the call.
- The number that the variable expands to during vector processing must be a valid entry for the command parameter.
- The number that the variable obtains during processing must be a valid announcement extension assigned on the Audio/Announcement screen.

Collect command

The following syntax rules apply when vector variables are used with the **collect** command.

```
collect [ced, cdpd] for [A-Z, AA-ZZ]
collect [1-16] digits after announcement [A-Z, AA-ZZ] for [A-Z, AA-ZZ]
```

Requirements and considerations

The requirements for using vector variables with the **collect** command are as follows:

- If `none` is specified after the for parameter, the **collect digits** command ignores the for parameter.
- The specification in the Variables tables defines what portion of the collected digits is assigned to the variable.
- The “#” digit can be collected and exist in the dial-ahead digits buffer if dialed by the caller. The “#” is assigned to a variable if that is the only digit assigned by the for parameter. This matches the threshold field with a “#” keyword.

  Example: If the caller dials “1#” and the specification for variable *B* starts at digit position 2 when length is more than 1, the single digit “#” is assigned to variable *B* by **collect 2 digits after announcement 1000** for the B command. If the dial-ahead buffer contains a “#” digit, the command **collect 1 digit after announcement 1001 for C** where C is defined as length = 1 and start = 1, then the “#” is assigned to variable C. A **goto step x if B = #** or **goto step x if C = #** is true and the branch to step x is taken. Also, the Variables for Vectors table shows the current value of “#” in the **Assignment** field.

However, you cannot assign a value of "#" to a variable using an entry in the **Assignment** field. You can only assign the "#" value to the variable using the **collect** … **for** command.

Converse-on command

The following syntax rules apply when you use vector variables with the **converse-on** command:

```
converse-on split [hunt group, 1st, 2nd, 3rd] pri [l, m, h, t]
passing [A-Z, AA-ZZ] and [A-Z, AA-ZZ]
converse-on split [hunt group]
pri [l, m, h, t]
passing [A-Z, AA-ZZ] and [A-Z, AA-ZZ]
```

Note:

A valid hunt group is an ACD skill or a non-ACD hunt group assigned for AUDIX, remote AUDIX, MSA, or QSIG Message Wait Indicator (MWI).

From Avaya Aura® Call Center Elite 7.1 onwards, vectoring supports the ability to pass the Last Agent Login ID, using the *agent* variable, as a parameter in the **converse-on** vector command in a VDN Return Destination (VRD) vector.

```
01 converse-on skill 801 pri l passing LA and none [LA - 'agent' variable]
```

Requirements and considerations

The requirements and considerations for using vector variables with the **converse-on** command are as follows:

- You can use a variable as a data type in both passing fields. This results in outpulsing the current value of up to 16 digits for each of the specified variables as a DTMF digit stream to the VRU or IVR connected by the **converse-on** command.
- The normal **converse-on** command rules for both passing fields apply. If the variable is defined, the passed DTMF digits are the current assignment of the variable followed by a # DTMF digit. If a variable is not defined, or assigned to none or #, a single # DTMF digit is outpulsed for that data item (treated as though the data type is none) and a vector event 38 (variable not defined) or vector event 213 (no digits in variable) is logged.

Disconnect command with vector variables

Communication Manager supports variable syntax for the **disconnect** command. You can use vector variables with the **disconnect** command after the announcement extension.

The following syntax rules apply when using vector variables with the **disconnect** command.

```
disconnect after announcement [A-Z, AA-ZZ]
```

Goto commands

The following syntax rules apply when you use vector variables with the **goto** command.

`goto` step 1-99 if **or** **`goto` vector 1-8000 @ step 1-99 if**					
A-Z, *AA-ZZ*	>,<, =, <>, >=, <=	*A-Z or AA-ZZ*			
	in table	*A-Z or AA-ZZ*			
	not in table				
ani	>,>=,<>,=,<,<=	*A-Z or AA-ZZ*			
	in table				
	not in table				
available-agents	in skill	hunt group, skills for VDN: 1st, 2nd, or 3rd	>,>=,<>, =, <,<=	*A-Z or AA-ZZ*	
calls-queued	in skill	hunt group, skills for VDN: 1st, 2nd, 3rd	pri l = low m = mediumh = high t = top	>,>=,<>, =, <,<=	*A-Z or AA-ZZ*
counted-calls	to vdn	vdn extension, latest, active	>,>=,<>,=, <,<=	*A-Z or AA-ZZ*	
digits	>,>=,<>,=,<,<=	*A-Z or AA-ZZ*			
	in table	*A-Z or AA-ZZ*			
	not in table				
expected-wait	for best	>,>=,<>,=,<,<=	*A-Z or AA-ZZ*		
	for call				
	for split	hunt group	pri l = low, m = medium, h = high, t = top	>,>=,<>, =, <,<=	*A-Z or AA-ZZ*
	for skill	hunt group, skills for VDN: 1st, 2nd, or 3rd			
holiday	in table	*A-Z or AA-ZZ*			
	not in table				
ii-digits	>,>=, <>, =, <, <=	*A-Z or AA-ZZ*			
	in table	*A-Z or AA-ZZ*			
	not in table				
interflow-qpos	>, >=, <>, =, <, <=	*A-Z or AA-ZZ*			

Table continues…

`goto` step 1-99 if **or** **`goto` vector 1-8000 @ step 1-99 if**					
oldest-call-wait	in skill	hunt group, skills for VDN: 1st, 2nd, or 3rd	pri l = low, m = medium, h = high, t = top	>, >=, <., =, <, <=	*A-Z or AA-ZZ*
rolling-asa	for skill	hunt group, skills for VDN: 1st, 2nd, or 3rd	>, >=, <., =, <, <=	*A-Z or AA-ZZ*	
staffed-agents	in skill	hunt group, skills for VDN: 1st, 2nd, or 3rd	>, >=, <>, =, ,, <=	*A-Z or AA-ZZ*	
V1-V9	>, <, =, <>, >=, <=	*A-Z or AA-ZZ*			
	in table	*A-Z or AA-ZZ*			
	not in table				
wait- improved for	best	>, >=, <>, =.<, <=			*A-Z or AA-ZZ*
	skill	hunt group, skills for VDN: 1st, 2nd, or 3rd	pri l = low, m = medium, h = high, t = top	>, >=, <>, =, <, <=	
	split	hunt group			

Requirements and considerations

The requirements and considerations for using vector variables with the **`goto`** command are as follows:

- A vector step that uses variable parameters displays command syntax similar to the following example, which tests the current number of counted calls for the active vdn to a user-defined variable “G”:

  `goto step 4 if counted-calls to vdn active <=G`

- Depending on the variable type that you use, the specifications that you provide for it, and the way in which you use it in a vector, the number of potential applications for vector variables is extremely large.

Route-to number command

The following syntax rules apply when you use vector variables with the **`route-to number`** command.

route-to number	up to 16 digits (0-9) <digits>[A-Z, AA-ZZ] <digits>*<digits>A <digits>#<digits>A <digits>~p<digits>A <digits>~m<digits>A <digits>~s<digits>A <digits>~w<digits>A <digits>~W<digits>A ~r<digits>A ~r+<digits>A	with cov `y or n`	if	digit	>, >=, <>, =, <=	`0-9` or `#`
				interflow-qpos	<, =, <=	`1-9`
				unconditionally		
<digits> notation for 0 or other digits in the range of 0-9. A vector variable [A-Z, AA-ZZ] can be entered. Also shown by an “A” notation. “~r” invokes Network Call Redirection (NCR) over the incoming trunk. “~r+” invokes NCR with E.164 numbering notation for incoming SIP trunking when required by the service provider.						

Requirements and considerations

The requirements and considerations for using vector variables with the **`route-to`** command are as follows:

- A variable can be used in the number field as the destination address for the **`route-to`** command. When the route-to number [A -Z, AA-ZZ] step is executed, the current numerical value or assignment of up to 16 digits is used for the destination. The variable is defined in the Variables for Vectors screen.
- A variable can be used in place of digits with all the possible special characters and digits entered before the variable.
- If the vector variable or resultant destination is not defined or is invalid, the route-to step fails, a vector event 38 (variable not defined) is logged, and vector processing continues at the next vector step. The destination number retrieved from the string of digits of the current value of the variable must be a valid destination as defined by the Communication Manager dial plan. Otherwise, the **`route-to`** command fails the vector event is logged, and vector processing continues at the next step.

Set command

The following syntax rules apply when you use vector variables with the **`set`** command:

```
set [variables, digits] = [operand1] [operator] [operand2]
```

The following fields can consist of vector variables.

Field	Allows the following vector variables
Variables	User-assigned *A-Z* or *AA-ZZ collect* type vector variable. The *collect* variable type can be global, local, or local persistent. Only *collect* variables can be assigned to by the **`set`** command. Others variable types can be used as the operands but cannot be assigned a value.
Operand1 Operand2	• User-assigned *A-Z* or *AA-ZZ collect* vector variable. The *collect* variable type can be either global, local, or local persistent. • System-assigned *A-Z* or *AA-ZZ* vector variables such as *ani*, *asaiuui*, and *doy*. From Avaya Aura® Call Center Elite 7.1 onwards, you can use the **`set`** command to set an *asaiuui* variable to the last agent login ID in a VRD vector. You can achieve this by adding the *asaiuui* variable to the left of the = operand and the *agent* variable to the right of the = operand, with the `SEL` operator and the length specified as operand2.

Wait command

The following syntax rules apply when you use vector variables with the **`wait`** command:

```
wait-time [0-999] sec hearing [A-Z, AA-ZZ]
[music, ringback, silence, continue]
```

VIV requirements

VIV works on all platforms and operating systems that are supported by Communication Manager.

The **MultiVantage G3 Version** field on the System-Parameters Customer-Options screen must have the following settings:

- The **Call Center Release** field must be set to `12.0` or later.
- The **Vectoring (Variable)** field must be set to `y`.

VIV interactions and considerations

Interactions and considerations	Description
Avaya CMS	Vector administration supports the vector variable command syntax on Avaya CMS. However, the definition of each variable can only be administered through Communication Manager on the Variables for Vectors administration table. **Note:** The specific commands for which variables are supported, depends on the CMS and Communication Manager release.
Variable failure conditions	When the tested variable conditional is not defined in the variables for vectors administration table, a **`goto`** test fails. The call does not branch and processing falls through to the next vector step.

Table continues…

Interactions and considerations	Description
Retention of vector variable values and assignments	The content of a local vector variable exists only while a call is in vector processing. Once a call exits vector processing, the value is cleared. Note that a call that experiences a **`converse-on`** vector command remains in vector processing. In addition, a **`route-to`** or **`adjunct routing link`** step that routes to a local VDN extension also remains in vector processing. Therefore, values that can be obtained by the call related local vector variables (*ani*, *asaiuui*, *collect*, *stepcnt*, *vdn*, and *vdntime*) and the value stored in “digits” can be used in a subsequent routed-to vector or vector steps for the same call. The value of a vector variable is not directly passed during an adjunct routing route request operation. The adjunct routing route request operation does pass the value of the “digits” buffer using the collected digits Information Element (IE). The vector **`set`** command can populate the “digits” buffer. The value determined by or assigned by using vector variables can be written to the “digits” buffer and become available to an adjunct. The **`set`** command can also be used to assign a value to the ASAI UUI string using the *asaiuui* vector variable type.
Vector Variable Usage Data	Enter the `list measurements summary` command to view the vector variable usage data on page 4 of the Measurement Summary report.

Definition of local, global, and local persistent variables

The variable type of a vector determines the scope of the vector variable. Based on the variable type, the scope can be global, local, or local persistent.

Local scope

When a variable has a local scope, the value of the variable is assigned on the basis of call-specific information and is applied only in the vector that currently processes the call.

For example, *asaiuui* variables always have a local scope. If variable *B* is administered as an *ASAI* variable and included in a vector step, variable *B* is assigned the unique ASAI user data value for each new call processed by the vector.

Local persistent scope

When the scope of a *collect* variable type is “local persistent”, that is, the P scope, the variable value is assigned on the basis of call-specific information and is applied in more than one vector that process the inbound call. Unlike an L scope local variable, wherein the value is valid only until the call is being processed by the current vector, the value assigned to the P scope local persistent variable persists until the call disconnects. Applications continue to use the P scope variable and the last assigned value when the call leaves an assigned VDN. The call is either returned to the assigned VDN through the VDN Return Destination (VRD) feature or is transferred by the answering agent to another VDN for further processing.

Note:

Use variables with local persistent scope to achieve objectives such as the following examples:

- Store the time of the day or the call duration. You can use the information to re-queue a call with higher priority if the call is transferred among many receivers.
- Track the last played announcement or the last entered collect digits. Collected digits store the number of attempts made by the customer for the life of a call.
- Count the number of VRD loops to force a call disconnection if the loop count exceeds a predetermined value. The count prevents calls from continuously reconnecting to the assigned VDN if the caller does not disconnect.

Global scope

Vector variables with a global scope have system-wide values that apply to all vectors in which the variables are used. For example, the value specified for a *tod* variable type is provided by the system clock. Despite the constantly changing value, at any given time the value is identical across all vectors that use the *tod* variable type.

For other variable types with a global scope such as *collect* or *value*, the assigned value is defined by a call center supervisor or an administrator. In this case, the user-defined value applies to all vectors that use the variable type.

About local variables

When you administer a variable with a local scope, the value assigned to the variable is provided from information that is specific to a call. Variable types that are local to the call or caller include *agent*, *ani*, *asaiuui*, *collect*, *stepcnt*, *vdn*, and *vdntime*.

Note:

ASAI data for a call can be modified by a CTI adjunct when a **`route-to`** adjunct command is used.

About global variables

- Some types of global variables require that you assign values. The value that you assign applies to all vector steps in which the variable is referenced, and all calls that are executing the vector steps. When you change the value, the change is reflected in all vector steps in which the variable is referenced. This rule applies to all global variable types that allow entry in the **Assignment** field on the Variables for Vectors screen or other methods to assign a value.

  Note:

  Some variable types allow you to use the **`set`** command, the collect digits step, an FAC or the active VDN to change the specified value. When you use any of the methods to change a variable value, the **Assignment** field on the Variables for Vectors screen updates to reflect the new variable value.

- Other types of global variables use dynamic system-retrieved values for which you cannot assign specific values. This rule applies to any global variable type that does not allow entry in the **Assignment** field of the Variables for Vectors screen such as the *tod* and *dow* variable types.

About local persistent variables

Local persistent variables have the same characteristics as local variables except that the assigned value for the *collect* type variable persists until the call disconnects.

System-assigned vector variable types

VIV provides different types of vector variables to meet various needs of call center operations.

Note:

As a call is processed through a vector or chain of vectors, the number of different variable types that can be applied is limited only by the type and number of variables that you administered.

System-assigned definition

This section describes the system-assigned vector variable types. The values for system-assigned vector variables come from the system. The values can come from any of the following methods:

- Communication Manager clock
- Data associated with a call such as asaiuui, and ani
- Processing of call such as stepcnt and vdntime

agent type variable

The *agent* type variable is assigned the agent identifier which is made available in a system variable for vectoring use when a customer call is redirected by the VDN Return Destination (VRD) feature into vector processing. The agent identifier can be included in User-to-User Information (UUI) using existing vector commands before routing the call to an external IVR system. By adding the agent identifier in UUI, post call surveys can furnish reports details down to the agent level.

The *agent* type variable is read-only and cannot be set. The *agent* variable can be used only in the following conditions:

- By setting an *asaiuui* variable in a VRD vector
- As a **`converse-on`** operand [`operand1` or *operand2*] in a VRD vector.

Note:

To use the *agent* type variable you must have Avaya Aura® Call Center Elite 7.0 or later.

Scope

The scope of *agent* type variable is always local.

ANI type variable

The *ANI* variable provides expanded testing of the caller's phone number. When you know who called, you can route the call based on the caller's area code, prefix, or suffix.

Scope

The scope for the *ANI* variable is local.

Example

The following vector example shows how you can use an *ANI* variable to determine the area code of the caller and then route the call to an office that shares the same area code. The following variable specifications are set on the Variables for Vectors screen.

Variable	Description	Type	Scope	Length	Start
A	Concatenates the area code of the caller	*ANI*	L	3	1
C	Destination	*Collect*	L	10	1

Variable A concatenates the incoming call to an area code. For example, if the calling ANI is 3035556002, A = 303. The call is routed to C, which is set to 3035381234.

```
1. ...
2. set C = A CATR 5381234 [C = 3035381234]
3. route-to number C
```

ASAIUUI type variable

The *ASAIUUI* variable is assigned a unique value for each incoming call based on ASAI user information. Once a value is assigned, the value can be modified by an adjunct after a **`route-to adjunct`** vector step. You can also use the **`set`** command to change the assigned value. A common use for an *ASAIUUI* variable in a vector step is to test the assigned value against a threshold value.

From Avaya Aura® Call Center Elite 7.1 onwards, you can use the *agent* type variable, by setting an *asaiuui* variable to the last agent login ID in a VRD vector.

Scope

The scope of *ASAIUUI* variables is only local.

Additional information

- Specify a start position for the *ASAIUUI* variable.
- Administer a length value for the *ASAIUUI* variable. Valid length values range from 1 to 16 digits, but if the digit length that extends from the specified start position to the end of the digit string is less than the specified length, the lesser number of digits is assigned. If the digit length that extends from the specified start position to the end of the digit string is greater than the specified length, any digits that extend the specified length are not included in the assigned value.

Example

The following example illustrates a vector step that compares an administered *ASAIUUI* variable D to a four-digit segment of the ASAI user information string that receives special call treatment if the first digit in the sequence is 3 and the last digit is 5:

```
goto step 5 if D = 3??5
```

Where D is an administered *ASAIUUI* variable and the threshold value that D is tested against is a four-digit string that begins with a 3 and ends with a 5.

Example

The following vector example illustrates how to use an *ASAIUUI* variable to provide selective customer treatment based on call-specific information.

In the example, a business wants to identify platinum member customers and provide special call treatment by queuing the customers at a higher level of priority. A CTI adjunct application uses the *ANI* data and other digits dialed by the caller to retrieve a five-digit customer account number. Account codes for platinum members are indicated by a 3 at the first digit position and a 5 at the last position in the five-digit string.

The adjunct includes the five-digit account number with other ASAI data beginning at digit position 4 in the 32-digit ASAI string.

Variable	Description	Type	Scope	Length	Start
P	Caller account code	*ASAIUUI*	L	5	4

The following example illustrates how to apply the administered *ASAIUUI* variable in a vector to implement the intended call treatment:

```
1. goto step 4 if P = 3???5
2. queue-to split 201 pri l
3. goto step 5 if unconditionally
4. queue-to split 201 pri m
5. announcement 3010
6. wait-time 30 secs hearing music
```

In the vector example, step 2 uses the *ASAIUUI* variable as a conditional value to test whether the account code for a call belongs to a platinum member (P = 3???5). If the caller is a platinum member, the call branches to step 4 where the call is placed in queue at a medium priority level. Otherwise, call control passes to step 2, which places the call in queue at a low priority level.

Example

The following vector example illustrates how to set an *ASAIUUI* variable to the last agent login ID in a VRD vector. By adding the agent identifier in UUI, post call surveys can furnish reports details down to the agent level. You can set an *asaiuui* variable to the last agent login ID by adding the *asaiuui* variable to the left of the = operand and the *agent* variable to the right of the = operand, with the `SEL` operator and the length specified as operand2.

```
01 set CH = LA  SEL  7     [CH - 'asaiuui' variable, LA - 'agent' variable]
```

DOW type variable

The *DOW* variable provides the current day of the week. The assigned value can range from 1 to 7, where 1 equals Sunday, 2 equals Monday, and so forth. The values assigned to this variable are obtained from the system clock on Communication Manager.

Scope

The scope for the *DOW* variable is global.

Example

In the following example vector step, if D is the *DOW* type variable, this step verifies that the day of week is in vector routing table 1.

```
goto step 2 if D in table 1
```

The vector routing table can have certain days of the week specified - for example, Sunday = 1 and Saturday = 7. If the variable D = 1 or 7, the goto step condition passes and goes to step 2. Otherwise, the *DOW* is a weekday Monday = 2 through Friday = 6 and the goto continues to the next step.

This example works similarly for day of year and time of day.

DOY type variable

The *DOY* variable provides the current day of the year. The assigned value can range from `1 to 366`. The 366 value is provided for leap years. The values assigned to the variable are retrieved from the system clock on Communication Manager.

Important:

Leap years include an extra day (February 29). Therefore, vectors that are initially set up in non-leap years, and include *DOY* variables with assigned values greater than 59 (February 28) must be shifted forward one day when a leap year begins. Alternately, when *DOY* variables are included in vectors that are initially set up in leap years, the variables must be shifted back one day when a non-leap year begins.

If a value of 366 is assigned to a *DOY* variable and the current year is not a leap year, any **`goto`** step in which the variable is used fails.

Scope

The scope for the *DOY* variable is global only.

Example

In the following example vector step, if D is the *DOY* type variable, the step verifies a day of the year.

```
goto vector 214 if D = 45
```

The example verifies that the day is Valentine's Day. January 31 plus February 14 equals 45. If the *DOY* is Valentine's Day, the call goes to vector 214. Otherwise, the call continues processing the next step.

Stepcnt type variable

The *stepcnt* variable tracks the number of vector steps. Before the number of vector steps reaches the maximum limit, the call can be rerouted instead of being dropped. The *stepcnt* variable can also be used as a loop-control variable. By monitoring the number of vector steps, you can:

- Reroute calls before the maximum limit for the system is reached and prevent calls from being dropped.
- Reroute calls after an action has reached a predetermined limit. For example, calls can be rerouted after an announcement or music has finished playing.

You can:

- Assign a variable between *A-Z*, or *AA-ZZ*.
- Assign the number of vector steps including the current step.
- Use the variable type anywhere other vector variables or VDN variables are used.
- Use the variable type as a threshold, conditional, or destination or data number, where supported.

Scope

The scope for the *stepcnt* variable is local.

Example

The following vector example illustrates how you to use a *stepcnt* variable to break out of a vectoring loop before a step limit is reached. The following variable specifications are set on the Variables for Vectors screen.

Variable	Description	Type	Scope
C	Sets the step limit	*stepcnt*	L

In step 6, if the system reaches more than 990 vector steps, an announcement plays to inform the customer about the high volume of calls.

```
1. wait-time 0 secs hearing ringback
2. queue-to skill 100 pri l
3. wait-time 10 secs hearing ringback
4. announcement 2000
5. wait-time 60 secs hearing music
6. goto step 8 if C >= 990
7. goto step 4 unconditionally
8. announcement 3000 [We are experiencing an unusually high volume of calls, please
leave your name and number for call back]
9. messaging skill 200 for extension active
```

Note:

Use a value that is less than the maximum number of vector steps, 10,000.

TOD type variable

The *tod* variable provides the current time of day based on a 24-hour format. The assigned value, which can range from 00:00 to 23:59, is received from the Communication Manager clock.

The values assigned to this variable are received from the system clock on the Communication Manager.

Communication Manager always returns four digits for the *tod* variable. This includes leading zeros where appropriate. Any comparison to the *tod* variable is also formatted as four digits. To check when the *tod* variable is after 12:30 a.m, compare to 00:30 and not 30.

Scope

The scope for the *tod* variable is global.

Example

In the following example vector step, if *D* is the *tod* variable type, this step verifies the current time of day.

`goto step` 32 if *D* >= 16:55

The example verifies that the time of day is 4:55 p.m. If the time of day is 5 minutes before closing, the call is routed to step 32. Step 32 can be an **`announcement`** step indicating that the call center is closed for the day.

VDN type variable

When a *vdn* variable is used in a **`goto`** step, the extension number value assigned to the variable is based on either the active or latest VDN associated with the call. The number of digits assigned to a *vdn* variable depend on the dial plan used for the system.

The latest value represents the VDN extension number associated with the vector currently in control of the call process and the active value represents the extension number of the current VDN, as defined by VDN override settings.

When administering the variable, you can specify whether to apply the active or latest value to *vdn* variables.

Scope

The scope for the vdn variable is only local.

Additional information

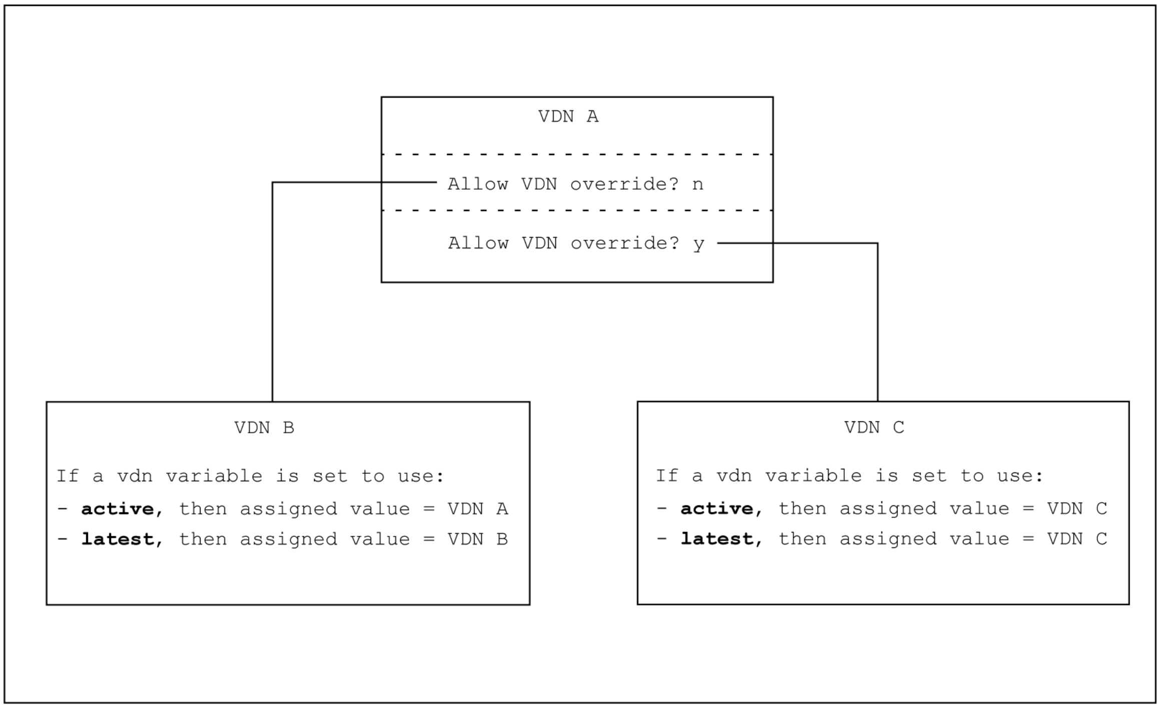
When a vdn variable is administered to use the *active* VDN of the current call as its value assignment, VDN override settings can affect the VDN extension number that is actually assigned to the variable.

When the **Allow VDN Override** field is set to `y` on the Vector Directory Number screen for a VDN, the extension number for the “subsequent” VDN to which a call is routed is applied to the call instead of the extension number for the current (latest) VDN. Therefore, the following rules apply for the value assigned to a vdn variable when it is used in a vector:

- If the VDN override setting for the previous VDN is not set to allow overrides, and a vdn variable in the vector associated with the next VDN in the call process flow is set to *active*, then the number for the previous VDN is assigned to the variable. An example of this case is represented in the following figure by the call flow from VDN A to VDN B.
- If the VDN override setting for the previous VDN is set to allow overrides, and a vdn variable used in the vector associated with the next VDN in the call process flow is set to *active*, then

the current VDN number is assigned to the variable. An example of this case is represented in the following figure by the call flow from VDN A to VDN C.

- When the vdn variable is set to use the *latest* VDN number, the VDN override setting for the previous VDN has no effect on the value that is assigned to the variable. This case is represented in both of the call flows shown in the following figure.



Example

The following example shows a goto vector step that uses an administered *vdn* variable G to execute a branching step when the VDN extension 4561 is identified:

`goto step` 5 if G = 4561

VDNTime type variable

The *vdntime* variable tests the time taken by the call center to process a call. This also includes any prior time spent in a remote Communication Manager. Administrators use the *vdntime* variable to determine when alternate routing, queuing, or call treatment is needed, based on the total time the call has been in the system.

When the *vdntime* variable is tested in a vector, a value is assigned that is equal to the number of seconds the call has been active in vector processing since the call first reached a VDN. If the processing started in a remote system which forwarded the call to this system using LAI or BSR, the time spent in the prior system is included.

Scope

The scope for the *vdntime* variable is local.

Example

The following vector example shows how you can use a *vdntime* variable to remove a call from a loop after 5 minutes. The following variable specifications are set on the Variables for Vectors screen.

Variable	Description	Type	Scope
T	Call processing time	*vdntime*	L

In step 5, if the T variable is greater than 300 seconds, or 5 minutes, the vector transfers control to step 1 in vector 289.

```
1. queue-to skill 51 pri l
2. wait 30 secs hearing ringback
3. announcement 1000
4. wait-time 60 secs hearing music
5. goto vector 289 @step 1 if T > 300
6. goto step 3 if unconditionally
```

Example

You can use the same approach, as in the previous example, with BSR Local Treatment vectors to break out the local wait treatment loop when the process time of the call exceeds the tolerable time period to take back the call and provide an alternative treatment. The example can be expanded for a call take back as follows:

```
change vector 40                                                     Page 1 of 3
                                    CALL VECTOR

Number: 40                    Name: Local BSR vector
                 Attendant Vectoring? n          Meet-me Conf? n           Lock? n
    Basic? y    EAS? y   G3V4 Enhanced? y       ANI/II-Digits? y   ASAI Routing? y
Prompting? y    LAI? y  G3V4 Adv Route? y        CINFO? n BSR? y       Holidays? y

01 announcement 3000
02 consider skill 4 pri m adjust-by 0
03 consider skill 6 pri m adjust-by 0
04 consider location 1 adjust-by 10
05 consider location 2 adjust-by 10
06 queue-to best
07 announcement 3001
08 wait-time 10 secs hearing music
09 goto step 11 if T > 300
10 goto step 7 if unconditionally
11 route-to number 54010 if unconditionally
```

User-assigned vector variable types

VIV provides different types of vector variables to meet various needs of call center operations.

Note:

As a call is processed through a vector or chain of vectors, the number of different vector variable types that can be applied is limited only by the type and number of vector variables that you have administered.

User-assigned definition

You can change the value of user-assigned vector variables. By contrast, the values for system-assigned vector variables are assigned from the system clock, data about the incoming call, or by the processing of the call.

Collect type variable

One of the ways that the *collect* type variable can be assigned a value is by using the **`collect`** command. When VIV is active on the server system, the **`collect`** command includes a for parameter that precedes the *collect* variable to which you can assign user dialed data to a *collect* type variable entered in this field.

Syntax

The basic syntax for the **`collect`** command when assigning a value to a variable “V” is shown in the following example vector step:

```
collect 2 digits for V
```

where V is a vector variable of type collect, as defined in the Variables for Vectors table.

Note:

Use of variables with **`collect`** command is not required. The default entry that follows the for parameter is none.

Other ways to assign a value to a *collect* type variable is by using the Variables for Vectors table (for the globe scope only) or by assignment using the **`set`** vector command. When used with the **`set`** command the *collect* type variable serves as a general purpose variable for implementing many different kinds of applications. An example using the administration table to assign a value of 14 to variable *V* is shown in the following example excerpt from the table.

Variable	Description	Type	Scope	Length	Start	Assignment	VAC
V	Local collect variable	*collect*	G	2	1	14	NA

Note:

Local or local persistent collect type variables can be assigned using the Variables for Vectors table.

Following is an example of using the **`set`** command to assign a value 14 to a variable *V*:

```
set V = 14 ADD none
```

A *collect* variable can also be used as a threshold value in a conditional test, as shown in the following example vector step:

```
goto step 4 if counted-calls to vdn active <=V
```

Scope

The scope of collect variables can be either local (L), local persistent (P), or global (G). The following rules apply:

- If the scope is local, the assigned value is null until a value is provided during processing for the call. The assigned value is retained through all further call processing steps, including any chained vectors and **`route-to`** VDN commands, until a new value is assigned during vector processing or initial vector processing for the call is terminated, at which time the value is cleared.
- If the scope is local persistent, the assigned value is null until the value is provided during processing for the call. Unlike a variable with local scope, a collect type variable with local persistent scope persists until the call disconnects. The application can continue to use the variable when the call leaves vector processing and then returns to vector processing, such as via VDN return destination or a subsequent transfer to another VDN by the answering agent. Persistent local collect variables retain the last value assigned during vector processing, and after termination of vector processing, internal transfer to another local vector, RONA/ROIF/ROOF return to vector processing or VDN Return Destination return to vector processing, until the call disconnects.
- If the scope is global, the assigned value is retained as a system-wide variable value until it is reassigned, either by changes made to the Variable for Vectors screen, or by a collect digits/ced/cpd for [A-Z, AA-ZZ] vector step designed for that purpose.

Additional information

- When collected data or other digit sequence is assigned to a *collect* variable, the value can be truncated by specifying a start position other than the first digit in the collected data string. A start position must be specified.
- A length value must be administered. Valid length values range from 1 to 16 digits, but if the digit length from the specified start position to the end of the digit string is less than the administered length value, the lesser number of digits is assigned. If the digit length that extends from the specified start position to the end of the digit string is greater than the specified length, then any digits that extend the specified length are not included in the assigned value.
- You can administer a local *collect* variable to persist until the call disconnects. This can be used, for instance, to pass collected digits if the call is transferred to another VDN or to serve as a counter for VDN Return Destination looping to cause the call to be disconnected after a certain number of iterations. Using the *collect* variable, you can limit the VDN Return Destination looping to disconnect the call if the caller does not hang up when required with use of **`set`** command to increment the variable each time the call is returned to the VDN Return destination VDN.

Example

You can use a *collect* variable to set a threshold value that controls how call center resources are allocated to different activities. In the following example, a call center wants to be able to adjust the amount of resources that are dedicated to a promotional sales giveaway campaign so that extra resources are shifted to more profitable sales campaigns during peak call volume hours.

In this example, a *collect* variable is used as a threshold to specify the number of calls allowed for the giveaway campaign, which is initially set to a value of 50.

The *collect* variable is applied as a threshold conditional in a counted-calls vector step in such a way that it can be quickly changed when reallocation of agent resources is necessary.

The specifications that you can provide on the Variables for Vectors screen for the *collect* variable used in this example are shown in the following table.

Variable	Description	Type	Scope	Length	Start	Assignment
G	Allowed calls for give-away campaign	*collect*	G	2	1	50

After the *collect* variable G is administered, you can create a vector that uses the variable as a conditional threshold. A counted-calls step that tests the variable conditional is shown in the following example vector.

```
1. wait-time 0 secs hearing ringback
2. goto step 4 if counted-calls to vdn active <=G
3. busy
4. queue-to skill 30 pri l
5. wait-time 10 secs hearing ringback
6. announcement 1002 [All agents are busy, your call is important.]
7. wait-time 60 secs hearing music
8. goto step 6 unconditionally
```

A second vector is administered so that the call center manager can quickly change the assignment for variable G. As shown in the following example, step 4 uses a **`collect digits`** command to allow an authorized user to change the number of calls allowed for the giveaway campaign.

```
1. wait-time 0 secs hearing ringback
2. collect 4 digits after announcement 10010 for none [Enter your security code]
3. goto step 7 if digits <> 1234
4. collect 2 digits after announcement 10011 for G [Enter the number of allowed active
calls.]
5. announcement 10012 [Your change has been accepted.]
6. disconnect after announcement none
7. disconnect after announcement 10013 [The security code you entered is incorrect.]
```

Example 2

You can use a *collect* variable with scope local persistent (P) in a vector to detect callers that do not hang up when required, and to disconnect their calls. In the following example, a call center has a VDN Return Destination assigned to the incoming call VDN named VDN1. After the agent completes the call, VDN1 forwards the call to VDN2, which in turn connects the call to a survey provided by a VRU device. Callers are required to disconnect the call when the survey is complete, but some callers can still remain connected.

With a local persistent *collect* variable, you can detect the callers for whom the call has already been connected to the survey once. To count the VDN returns, you can set up a vector using the collect variable with scope local persistent.

The specifications that you can provide on the Variables for Vectors screen for the *collect* variable used in this example are shown in the following table.

Variable	Description	Type	Scope	Length	Start	Assignment
C	VDN return counter for the calls	collect	P	1	1	NA

After *collect* variable P is set up, administer C to be 0 using the following step in the vector assigned to VDN1:

```
set C = none ADD 0
```

Administer the vector assigned to VDN 2 to check the count for the P scope collect type variable C and disconnect the call if the count has reached 1. In the below vector, step 3 tests the value of C. If the value of C is greater than 0, which means the call had already been connected to the survey, processing for the call is branched to step 7. Step 7 disconnects the call without playing an announcement. Step 4 adds 1 to the collect variable C since the call is processed by the survey announcement in the next step.

```
1. wait-time 0 secs hearing ringback
2. goto step 4 if counted-calls to vdn active <=G
3. goto step 7 if C>0  [C was initialized to 0 by the vector assigned to VDN1]
4. set C = C ADD 1
5. step for connecting the caller to the survey
6. stop
7. disconnect after announcement none
```

Value type variable

With the *value* variable type, you can change vector applications from one operational mode to another. To implement value variables, perform the following steps:

- Administer a value variable in the variable administration table.
- You can administer an FAC and associate the FAC with a Variable Access Code (VAC) to use a dial code procedure to change a variable value assignment. VAC designations VV1 through VV9 are provided on the FAC screen.
- If you associate an administered *value* variable with a FAC, you can dial the FAC and enter a single digit (0 to 9) to change the variable assignment. If the variable is not associated with a FAC, you must change the variable assignment in the Variables for Vector screen.

Scope

The scope of *value* variables is global.

Reason to use

One of the reasons to use the *value* variable is to change vector processing through a manual operation. Without value vector variable, call centers have to use a dummy agent logged into a dummy skill to detect the status of a call center such as a disaster event or a closure. Now the value vector variable type can be used as the trigger that can be tested in vectoring using a **`goto vector`** command. The trigger can be set by dialing the FAC assigned to the value type variable and entering in the value that changes the vector processing for the calls to that vector. For example, *value* variable *V* can be set to 0 for normal operation and then to 1 to trigger the disaster operation using the FAC.

Additional information

- You can use the phone to access an FAC if you associate a *value* variable with an FAC. You can also change the assigned variable value. If you do not create an FAC to use with a *value* variable, the only way to change the assigned variable value is to change the **Assignment** field in the Variables for Vectors table.
- If you set up an FAC to change a value variable assignment, a station user must use a physical phone that has the required console permissions set to yes on the Class of Service screen.
- To reset the assigned value for a value variable to `null`, access the FAC associated with variable and enter `*` instead of a digit.

Example

The following example shows how to use the *value* variable *A* as a conditional in a vector step:

`goto vector` 34 if A = 2

where *A* is an administered *value* variable and the value that *A* is tested against is an arbitrary, single-digit number that represents an operational mode or condition for response in your call applications.

VIV administration

Note:

For most variable types, administration is done solely in the variables administration table. However, a Feature Access Code (FAC) administration step is also required to use an FAC to change assignments for value variables.

Example Variables for Vectors screen

Use the following screen to administer vector variables.

```
change variables Page 1 of x
                                 Variables For Vectors
 Var  Description                  Type    Scope Length Start Assignment       VAC
 A    ___________________________ _______ _     __     __    ________________ ___
 B    ___________________________ _______ _     __     __    ________________ ___
 C    ___________________________ _______ _     __     __    ________________ ___
 D    ___________________________ _______ _     __     __    ________________ ___
 E    ___________________________ _______ _     __     __    ________________ ___
 F    ___________________________ _______ _     __     __    ________________ ___
 G    ___________________________ _______ _     __     __    ________________ ___
 H    ___________________________ _______ _     __     __    ________________ ___
 I    ___________________________ _______ _     __     __    ________________ ___
 J    ___________________________ _______ _     __     __    ________________ ___
 K    ___________________________ _______ _     __     __    ________________ ___
 L    ___________________________ _______ _     __     __    ________________ ___
 M    ___________________________ _______ _     __     __    ________________ ___
 N    ___________________________ _______ _     __     __    ________________ ___
 O    ___________________________ _______ _     __     __    ________________ ___
```

Required variable administration entries

The following table summarizes the information required in the various fields of the Variables for Vectors screen for the different types of variables.

Variable type	Scope	Length	Start	Assignment	VAC
agent	Local only (L)	—	—	The *agent* type variable is read-only and cannot be set. The *agent* variable can be assigned only in the following conditions: • By setting an *asaiuui* variable in a VRD vector • As a **converse-on** operand [`operand1` or *operand2*] in a VRD vector.	—
ani	Local only (L)	1 to 16 digits (required)	start position from 1 to 16 (required)	—	—
ani	Local only (L)	1 to 16 digits (required)	start position from 1 to 16 (required)	—	—
asaiuui	Local only (L)	1 to 16 digits (required)	start position from 1 to 96 (required)	—	—
collect	Local, Local Persistent, or Global (L, P, or G, required)	1 to 16 digits (required)	start position from 1 to 16 (required)	Local and Local Persistent- not applicable Global - 1 to 16 digits	—
dow	Global only (G)	—	—	—	—
doy	Global only (G)	—	—	—	—
stepcnt	Local only (L)	—	—	—	-

Table continues…

Variable type	Scope	Length	Start	Assignment	VAC
tod	Global only (G)	—	—	—	—
value	Global only (G)	1	—	1 digit (0 to 9, optional). If you do not assign a value in this field, a null value is specified. However, if you administer an FAC to set the variable assignment, any value that you assign by dial code procedure is subsequently displayed in this field.	VVX (optional). You must enter a VAC value to use an FAC to change the variable assignment. The format for the VAC value is VVX, where *X* is a single digit that ranges from 0 to 9. The VVX value that you list in this field, must be obtained from the FAC screen after you set up the FAC. In the FAC screen, the VVX value is displayed on the same line as the FAC code. If you do not specify a VVX value when you administer the variable, you will receive an intercept tone when you attempt to dial the FAC.
vdn	Local only (L)	—	—	active or latest	—
vdntime	Local only (L)	—	—	—	—

Performing optional FAC administration for value variables

About this task

This section describes the administration steps to use value variables in vectors and to use an FAC to change the variable assignments.

Use the following screen to administer an FAC.

```
change feature-access-codes                                 Page x of x

                          FEATURE ACCESS CODE (FAC)

                        Call Vectoring/Call Prompting Features

   Converse Data Return Code: ____

Vector Variable 1 (VV1) Code: ____
Vector Variable 2 (VV2) Code: ____
Vector Variable 3 (VV3) Code: ____
Vector Variable 4 (VV4) Code: ____
Vector Variable 5 (VV5) Code: ____
Vector Variable 6 (VV6) Code: ____
Vector Variable 7 (VV7) Code: ____
Vector Variable 8 (VV8) Code: ____
Vector Variable 9 (VV9) Code: ____
```

To administer an FAC, perform the following steps:

Procedure

1. On the Call Vector/Call Prompting Features page of the Feature Access Code screen, enter an FAC code in the field next to one of the Vector Access Code (VAC) entries.

   The FAC code must be a 1 to 4 digit string, but either a pound (“#”) or an asterisk (“*”) can be substituted for a numeral at the first digit position.

2. Note the VVX value associated with the new FAC code.

   Possible VAC entries range from VV1 to VV9. You must enter this value in the **VAC** field on the Variables for Vectors screen when you administer the value variable to be associated with the FAC.

VIV job aid

The following table summarizes the basic functions and characteristics of the different VIV types:

Variable type	Description	Scope	Specification	Max digit length	Assigns
agent	Maintains the identifier of the “Last EAS Agent” to drop off the call throughout the life cycle of the call. The lifecycle of the call can include Transfers, Blind or Consultative, conferencing, or re-routing. The *agent* type variable is read-only and cannot be set. The *agent* variable can be assigned only in the following conditions: • By setting an *asaiuui* variable in a VRD vector • As a **`converse-on`** operand [`operand1` or *operand2*] in a VRD vector.	L			Identifier of the “Last EAS Agent” to drop off the call throughout the life cycle of the call.

Table continues…

Variable type	Description	Scope	Specification	Max digit length	Assigns
ani	Holds the phone number of the caller	L	Start digit position and Length	16	Incoming call data
asaiuui	Holds call-specific user data associated with the caller	L	Start digit position and Length	16 out of a total of 96	Incoming call or ASAI application data
collect	Holds user-defined digits associated with the call for control, routing or special treatment that can be assigned a value by the Variables for Vectors table, collect digits steps or the **`set`** vector command.	L P G	Start digit position and Length	16	The for parameter of the **`collect digits`** command or assignment in the variables table
tod	Holds the current time of day in 24-hour format for processing	G	None	Always 4	The main server system clock - for example, 02:19 = 02:19 am
dow	Holds the current day of week for processing	G	None	1	The main server system clock (1-7) - for example, 1 = Sunday
doy	Holds the current day of year for processing	G	None	Always 3	The main server system clock (1-365 or 1 -366 in a leap year)
stepcnt	Provides the count of vector steps executed for the call, including the current step	L	None	4	The vector processing step counter
value	Holds a single numerical digit (0-9) for user-defined processing	G	None	1	A user-defined value entered using the VAC FAC procedure or assignment in the variables table
vdn	Holds the VDN extension number of the call for processing	L	Active or Latest VDN	16	Routing for a call
vdntime	Provides the time taken, in seconds, by a call center to process a call.	L	None	Always 4	Time in vector processing including prior processing for a call routed by BSR/LAI

VIV vector examples

This section includes simple examples that show how vector variables can be used to help improve call processing operations.

Example application using time and day variables

- You can use *tod* and *dow* variables to create flexible vectors that evaluate factors, such as hours and days of week, so an appropriate call treatment is delivered to customers.
- You can use global *collect* variables to define call center start and close times for different days of the week. The *collect* variables provide threshold values that are tested against *tod* and *dow* values to determine appropriate call treatments.
- You can set up special VDNs to reassign variable values for the opening and closing time. For instance, when a change in daylight saving time occurs. The new variable values are instantly propagated to any number of vectors in which the variables are used.

Scenario details

The example call center has the following daily hours of operation that must be specified in the 24-hour clock time:

Day of week	Opening time	Closing time
Monday to Thursday	0700	2300
Friday	0700	2100
Saturday and Sunday	0700	1600

How to administer the variables

The specifications that you can provide on the Variables for Vectors screen for the variables used in this example are shown in the following table.

Variable	Description	Type	Scope	Length	Start	Assignment
tod or *dow* variables						
T	Current time of day	*tod*	G			Obtains the current time of day from the system clock in 0000 - 2359 format
D	Current day of the week	*dow*	G			Obtains the current day of week in 1- 7 format (1=Sunday)
Start time or close time variables						
O	Opening time, all days of week	*collect*	G	4	1	0700

Table continues…

Variable	Description	Type	Scope	Length	Start	Assignment
L (In the current example, Monday through Thursday closing time defines an upper bound on the latest possible closing time for any day of the week. Therefore, variable designation L is used to signify the latest possible closing time.)	Closing time, Monday through Thursday	*collect*	G	4	1	2300
F	Closing time, Friday	*collect*	G	4	1	2100
W	Closing time, Saturday and Sunday	*collect*	G	4	1	1600

How to create a vector to use the TOD and DOW variables

The following vector example explains how the Time of Day (TOD) and Day of the Week (DOW) variables can be used for call center business hours to control call processing in a desired manner.

```
 1. goto step 30 if T < O                     [if tod is earlier than 0700 hours, go to
the out of hours treatment]
 2. goto step  8 if T < W                     [if tod is earlier than 1600 (earliest
possible closing time), working hours apply.
                                                        Continue with step 8]
 3. goto step 30 if D = 1                      [if dow is Sunday, go to the out of
hours treatment]
 4. goto step 30 if D = 7                     [if dow is Saturday, go to the out of
hours treatment]
 5. goto step  8 if T < F                     [if tod is earlier than 2100 (Friday
close time), working hours apply]
 6. goto step 30 if D = 6                     [if tod is later than 2100 (as determined
by the preceding step), and dow is Friday,
                                                        go to the out of hours
treatment]
 7. goto step 30 if T > L                     [if tod is later than 2300, go to the out
of hours treatment]
 8. goto step 31 if holiday in table 8  [based on the outcome of all preceding steps,
working hours apply unless today is a
                                                        holiday]
 9.  announcement 16549 [Please wait for the next available agent]
 10. consider skill 80 pri m adjust by 0
 11. consider location 16 adjust by 10
 12. queue-to best
 13. goto step 30 if staffed agents in skill 80 = 0
 14. wait-time 2 secs hearing silence
....
```

```
....
30. announcement 18465 [Please call again during regular business hours]
31. closed for holiday treatment
```

In the vector example, *tod*, *dow* and global *collect* variables control the call process flow by testing the call time and day values against a series of time windows that represent possible ranges of operational hours for the call center.

Steps 1 and 2 determine whether the time is within the minimum window of operational hours common to all work days, which is currently defined as 0700 to 1600 hours.

Step 1 tests whether the time is earlier than the 0700 opening time that is common to every day of the week (T < O). If the time is earlier than 0700, vector processing branches to the out of hours treatment at step 30. Otherwise, control passes to step 2.

Step 2 tests whether the time is earlier than the earliest possible closing time for any day of the week, which is 1600 on weekend days (T < W). If so, the call time is within the range of work hours that are common to all days of the week, and processing branches to step 8, which checks for a holiday before processing goes through the series of consider and queue-to best steps that are included in steps 9 through 12. Otherwise, vector processing goes to step 3 for further assessment.

Steps 3 and 4 then test whether the current day is Saturday (dow = 7) or Sunday (dow = 1). When either case is true, call control passes to the out of hours treatment provided at step 30. Otherwise, the call control passes to step 5 for further assessment.

Step 5 tests whether the time is earlier than the Friday closing time (T < F). If so, the current time is within the normal range of operating hours for Monday through Friday, and call processing branches to steps 8 through 12 for in-hours treatment. Otherwise, call vectoring goes to step 6 for further assessment.

Step 6 tests whether the day is Friday (dow = 6). If so, processing goes to out of hours treatment at step 30. Otherwise, call vectoring continues at step 7.

Step 7 completes the assessment of possible time windows by testing whether the tod is later than the latest possible closing time of 2300 hours on Monday through Thursday (T < L). If so, the call is directed the out of hours treatment provided at step 30. Otherwise, the time falls within normal work hours for Monday through Thursday and processing goes to steps 8 through 12 for in-hours treatment.

How to create a vector to reassign the hours of operation

tod and *dow* variables can be tested against *collect* variables that specify call center opening and closing time for different days of the week. Because global *collect* variables are used to specify the hours of operation, you can create a simple vector that allows the hours of operation to be changed very quickly and which is instantaneously propagated to multiple vectors.

The following example shows a vector that allows the call center opening time, which is specified by variable O in the current example, to be quickly changed by dialing a VDN dedicated for that purpose.

Note:

You must create other vectors like this one for each of the global *collect* variables that you use to set call center opening and closing time.

```
VDN 1
1. wait-time 0 secs hearing ringback
2. collect 5 digits after announcement 17000 for none  [Please enter your security
code.]
3. goto step 6 if digits <> 12345
4. collect 4 digits after announcement 17001 for O     [Please enter your daily opening
time.]
5. disconnect after announcement 17006                 [Your change is accepted.]
6. disconnect after announcement 17010                 [You have entered an invalid
security code.]
```

Example application using a value variable

The *value* variable always has a global scope and is designed to work with FACs so that the variable assignments can be quickly changed. One application of the *value* variables is to allow multiple call applications to be quickly switched from one operational mode to another. Such a rapid switch over capability can be useful for businesses whose operations are impacted by unpredictable events. For example, a public utility can desire a switch over capacity to respond to widespread power outages associated with severe weather events.

To set up a value variable to use in multiple vectors to meet such a special switch over need, you can administer both a value variable and an associated FAC, as described in the related topics.

How to administer an FAC to use with a value variable

For this example, the FAC code is accessed when you dial `*23`. The following administration screen shows how to enable an FAC.

Note:

When you administer the FAC for a variable, note the VVX number associated with the new FAC. The VVX value must be provided in the **VAC** field on the Variables for Vectors screen, as described in *administering the value variable*.

```
change feature-access-codes                                Page x of x

                      FEATURE ACCESS CODE (FAC)

                    Call Vectoring/Call Prompting Features

   Converse Data Return Code: ____

Vector Variable 1 (VV1) Code: _*23
Vector Variable 2 (VV2) Code: ____
Vector Variable 3 (VV3) Code: ____
Vector Variable 4 (VV4) Code: ____
Vector Variable 5 (VV5) Code: ____
Vector Variable 6 (VV6) Code: ____
Vector Variable 7 (VV7) Code: ____
Vector Variable 8 (VV8) Code: ____
Vector Variable 9 (VV9) Code: ____
```

How to administer the value variable

After you set up an FAC to use with the *Value* variable, you must administer the Variables in Vectors screen to set up the value variable associated with the FAC.

Variable	Description	Type	Scope	Length	Start	Assignment	VAC
S	Switch over for blizzard	value	G	1		1	VV1

In the variable administration table, verify that the VAC code has the same value that appears with the code number on the FAC screen. If a VAC entry is not provided, you will receive an intercept tone when you dial the FAC.

How to use the value variable in multiple vectors

After you complete the required administration for the *value* variable and its associated FAC, you can use it to redirect calls from vectors used for normal operational treatments to special treatment vectors that address the switch over conditions.

The following vector step can be used in multiple vectors to implement the change in operational mode:

```
goto vector 123 @step 1 if S = 2
```

In this example, the default value for the switch over variable is administered with a value assignment of “1”, to denote normal operational modes. When a switch over due to blizzard conditions is required, the call center administrator dials *23 to access the FAC and enters the digit “2” to indicate that switch over conditions are now in effect.

Example applications using global collect variables

This section presents VIV examples using a global collect variable type instead of the single digit value variable type. The value variable allows an FAC assignment so that the “value” of the variable can easily be changed by a dialing sequence. The global collect variable can be used in the same way, that is, by dialing a VDN instead of an FAC.

Global collect variables offer many advantages over the value variable, including:

- Global collect variables are not limited to nine FACs.
- The ability to add a security code to prevent malicious or accidental changes to the call flows.
- Creation of a VDN/vector menu that prompts for the variable to change (multiple variables via one VDN).
- Call flow routing can be changed remotely by calling the VDN rather than through remote access to dial an FAC.
- Additional features such as feedback announcements and confirmation.

How to verify a password and change a value

In the example, the global collect variable *A* is used in call center vectors to control the flow that a call experiences. The variable is assigned using the `change variable` command and is given a type *collect* and a scope of “G” for global. The other settings depend on how the variable is used. The length is 1 to 16 with a start position of “1”.

The vector prompts the user for a 16–digit password. The password is compared to the expected value contained on the active VDN as VDN variable “V2”. Step 8 prompts the user to enter new value for variable *A*. The announcement is recorded to list the expected values along with the use of each value.

The following example illustrates one approach to accomplish the tasks. The vector flow can also be written using vector subroutines for entry confirmation.

```
01 wait-time 2 secs hearing ringback
02 # Entry and Validation of Security Code
03 collect 16 digits after announcement V1 for none
04 goto step 6 if digits = V2
05 disconnect after announcement none
06 # This vector modifies the value of Variable A for global call flow
07 # control. Enter 111-Normal Ops, 222-Evacuation, 333-Severe Impairment
08 collect 3 digits after announcement V3 for A
09 # Goodbye
10 stop
```

How to add change confirmation

This example adds the following to the previous example:

- Announcement of the entered value.
- Change verification.
- Retry loop.

For this example, variable A is defined as a global collect variable of length 3 and a start of 1, and variable B (used as a temporary variable) is defined as a local *collect* variable of length 3 and a start of 1.

```
01 wait-time 2 secs hearing ringback
02 # Entry and Validation of Security Code
03 collect 16 digits after announcement V1 for none
04 goto step 6 if digits = V2
05 disconnect after announcement none
06 # This vector modifies the value of Variable A for global call flow
07 # control. Enter 111-Normal Ops, 222-Evacuation, 333-Severe Impairment
08 # or Enter 000 to exit
09 collect 3 digits after announcement V3 for B
10 goto step 32 if B = 000
11 # Play announcement to inform user what value they entered.
12 goto step 15 if B <> 111
13 announcement 61111
14 goto step 24 if unconditionally
15 goto step 18 if B <> 222
16 announcement 61112
17 goto step 24 if unconditionally
18 goto step 21 if B <> 333
19 announcement 61113
20 goto step 24 if unconditionally
21 # Non-valid digit string was entered announcement, please try again
22 announcement 61114
23 goto step 9 if unconditionally
24 # Please confirm that this is the desired value 0-no, 1-yes
25 collect 1 digits after announcement 61115 for none
26 goto step 30 if digits <> 1
27 set A = B ADD none
28 # Play announcement that value was changed and then disconnect.
29 disconnect after announcement 61116
30 # Value was not confirmed or incorrect - try again
```

```
31 goto step 9 if unconditionally
32 disconnect after announcement none
```

Example applications using vdn type variables

Use the *vdn* variable type to reduce the number of vectors required to provide differential treatment to specific service VDNs. The following examples show different ways to use *vdn* type variables to create a single vector that can be used by multiple VDNs, even as you maintain the ability to provide differential call treatment based on VDN identity.

The following table shows the specifications that you must enter on the Variables for Vectors screen for a *vdn* type variable that are used in the vector examples in this section.

Variable	Description	Type	Scope	Length	Start	Assignment
Y	VDN for DNIS testing	vdn	L			active

This first example shows how the administered *vdn* type variable *Y* can be used in a single vector to provide multiple announcement treatments based on call identity. Vector processing for the call proceeds through a series of paired **goto** and **announcement** steps that attempt to match the call VDN with an appropriate announcement.

```
1. goto step 3 if Y <> 1001
2. announcement 2001
3. goto step 5 if Y <> 1002
4. announcement 2002
5. goto step 7 if Y <> 1003
6. announcement 2003
7. goto step 9 if Y <> 1004
8. announcement 2004
9. queue-to skill 50
```

In step 1, the call-specific value for the *vdn* type variable *Y* is compared to one of several possible administered VDN values (Y <> 1001). If the value for *Y* matches the specified VDN value, an announcement treatment specific to that VDN is provided in step 2. Otherwise, vector processing branches from step 1 to the next test or announcement pair and proceeds until the caller receives an appropriate announcement treatment.

The following example shows another way that the *vdn* type variable can be applied in a vector to implement selective call treatment. In this example, the *vdn* type variable assigned to the call is tested against a VDN to distinguish local and non-local callers.

```
1. wait 0 secs hearing ringback
2. goto step 4 if Y = 4561  [VDN for 800 number callers]
3. announcement 2700  [Our store is located at 1300 West 120th Avenue.]
4. queue-to skill 30 pri l
5. wait-time 5 secs hearing ringback
6. announcement 1002  [All agents are serving other customers, please wait.]
7. wait-time 60 secs hearing music
8. goto step 6 if unconditionally
```

As shown above, step 2 tests whether the value assigned to the *vdn* type variable is equal to the VDN associated with 800-number callers (Y = 4561). If so, call control branches to step 4. Otherwise, call control passes to step 3, which provides an announcement intended specifically for local callers.

Example application using a vector variable in other commands

A vector variable can be used in the **route-to** number command to route a call to a destination provided indirectly through user input (collect type), a vdn type (active or latest VDN extension for the call) or from ASAI UUI (user data) associated with the call. This example uses a destination address provided remotely by ASAI UUI included with the call. The variable R defines the portion of the ASAI UUI digit string to use as the **route-to** number.

Variable	Description	Type	Scope	Length	Start
R	Alternate route to destination	asaiuui	L	5	3

You can use the *asaiuui* type variable R in a vector to route the call to the destination defined by a remote location if the number of staffed agents is less than a certain number. If the number of staffed agents is less than 100, the call is routed to the 5-digit destination indicated in the ASAI UUI, forwarded with the call from the remote location. Otherwise, the call must be put in queue for handling at the current location.

```
1. wait-time 0 secs hearing ringback
2. goto step 8 if staffed-agents in skill 22 < 100
3. queue-to skill 22 pri l
4. wait-time 6 secs hearing ringback
5. announcement 2001
6. wait-time 60 secs hearing music
7. goto step 3 unconditionally
8. route-to number R
9. goto step 3 unconditionally
```

At step 8, the variable R is assigned 5 digits of the call's ASAI UUI data digit string starting from digit position 3. This 5-digit number is used as the destination for the **route-to** command. Step 9 provides backup in case the **route-to** number command fails due to an empty ASAI UUI digit stream or the number obtained is an invalid destination.

From Avaya Aura® Call Center Elite 7.1 onwards, you can use the *agent* type variable, by setting an *asaiuui* variable to the last agent login ID in a VRD vector. By adding the agent identifier in UUI, post call surveys can furnish reports details down to the agent level. You can set an *asaiuui* variable to the last agent login ID by adding the *asaiuui* variable to the left of the = operand and the *agent* variable to the right of the = operand, with the `SEL` operator and the length specified as operand2.

```
01 set CH = LA  SEL  7     [CH - 'asaiuui' variable, LA - 'agent' variable]
```

Example application using a vector variable in the converse-on command

Including a vector variable in the **converse-on** command as a data item to pass (outpulse) to the VRU or IVR allows forwarding of additional data that is not currently a supported data type. You can define the variable as any of the existing variable types, such as *collect*, *value*, *tod*, *doy*, *dow*, and *asaiuui*. You can use the *asaiuui* type to forward data provided by a remote site or local ASAI interfaced application or with assignment using the **set** command. For this example, variable D forwards numerical account code data of up to 6 digits provided by an ASAI application.

Variable	Description	Type	Scope	Length	Start
D	ASAI provided data	asaiuui	L	6	1

The ASAI application uses adjunct routing to reach VDN2 that is assigned to the following vector. The data is included as ASAI UUI in the route-select message that routes the call to VDN2. The VRU interfaced through the **converse-on** command performs further interactive processing of the call based on the account code provided in the ASAI UUI and indicates where to next route the call.

```
1. wait-time 0 secs hearing music
2. converse-on skill 30 pri l passing vdn and D
3. collect 5 digits after announcement none for
4. route-to digits with coverage y
```

The **collect** command at step 3 collects the 5-digit destination provided by the VRU using the data return function. Step 4 routes the call to that destination.

From Avaya Aura® Call Center Elite 7.1 onwards, vectoring supports the ability to pass the Last Agent Login ID, using the *agent* variable, as a parameter in the **converse-on** vector command in the VDN Return Destination (VRD) vector.

```
01 converse-on skill 801 pri l passing LA and none [LA – 'agent' variable]
```

VDN In a Coverage Path

VICP

VDN in a Coverage Path (VICP) enhances call coverage and call vectoring. If you administer **Call Vectoring (Basic)** or **Call Prompting**, you can assign a VDN as the last point in a coverage path. A vector or Call Prompting processes calls that reach this coverage point.

VICP considerations

- A VDN is not allowed to be a member of a coverage answer group.
- A coverage answer group can only be a point in a coverage path. A vector cannot route a covered call to a coverage answer group.
- Removing a VDN from the system with the **remove vdn** *<extension>* command automatically removes the VDN from any coverage paths.
- By default, features such as Call Coverage, Call Forwarding, RONA, or Night Service cannot redirect calls that cover to a VDN. Therefore the **route-to digits** or **route-to number** command with **cov** set to `y` is treated the same as **cov** set to `n`.
- The **Cvg Enabled for VDN Route-to Party** field on the Coverage Path screen allows a second redirection (if needed) when a **route-to** command with cov = y vector step processes a previously covered call routed to a VDN. The default is `n` which retains the single redirection operation. If you set the field to `y` and the call has covered only once,

the call redirects again as though the call was placed directly to the VDN. For example, the following second redirection can occur:

1. the destination has a coverage path or another redirection feature.
2. the calling user presses the **go-to-cover** button while the destination rings the principal phone on the second coverage path. The call stops ringing at the second principal phone and moves to the next coverage path.
3. the second principal phone's **Hunt-to Station** path is checked for possible hunt before coverage or hunt after coverage redirection.
4. send-all-calls at the destination station that go to coverage applies.

VICP interactions

Interaction	Description
AAR/ARS Partitioning	The Class of Restriction (COR) assigned to the VDN determines the Partition Group Number (PGN). The PGN in turn determines the AAR/ARS routing tables used by **`route-to`** commands.
ASAI	For direct calls to a VDN, the **`adjunct routing`** command operates like the **`route to digits`** command with coverage set to `y`. For calls that cover to a VDN, the **`adjunct routing`** command operates like the **`route to digits`** command with coverage set to `n`.
Attendant	Vectors connect calls, covering to a VDN, to an attendant queue or a hunt group. Internal calls that route to an attendant display the COR of the originating station if the attendant presses the **display COR** button. An attendant cannot establish a conference with a call covering to a VDN if the call is in vector processing. If a call placed to a local destination has covered to a VDN and the attendant attempts to add this call to a conference, conferencing is denied until the call has completed vector processing. If the attendant extends a call to a local destination that covers the call to a VDN, the attendant’s return call timer cancels when vector processing begins and the **Return Call** button does not affect the call. If a call covers to a VDN and is then routed to an attendant, the attendant can transfer the call to another VDN.
AUDIX	Calls that cover to a VDN can be routed to an AUDIX by the **`route-to`** or **`messaging`** vector commands. Calls that cover to a VDN can be subsequently transferred to the AUDIX. Calls can also be transferred out of an AUDIX to a VDN.
Automatic Call Distribution (ACD)	A VDN can be the last point in an agent’s coverage path for direct agent calls.
Call Coverage	A VDN cannot be a member of a coverage answer group. A vector cannot route a covered call to a coverage answer group. Coverage Callback and Leave Word Calling work normally when a vector delivers a call to a covering user.

Table continues…

Interaction	Description
Call Forwarding	Calls that have covered to a VDN cannot be redirected by Call Forwarding unless the **Cvg Enabled for VDN Route-To Party** field applies.
Call Park	A parked call does not cover to a VDN. When a call is parked at an extension with a VDN in its coverage path, the call continues to ring the extension. If the call parks to a hunt group extension and the call is in queue, the call remains in the queue until it is retrieved, answered by an agent, or abandoned by the caller. A vector event is generated for the calls when the administered coverage criteria are met. Once a call covers to a VDN, Call Park cannot be established until the call is delivered to an extension and vector processing ends.
Call Vectoring	The COR assigned to a VDN determines the PGN. The PGN in turn determines the AAR/ARS routing tables used by **`route-to`** commands. When a call covers to a VDN, VDN Override has no effect on the display shown on an answering display. This station shows the normal display for a covered call. Adjunct Routing: For direct calls to a VDN, the **`adjunct routing`** command operates like the **`route to digits`** command with coverage set to `y`. For calls that cover to a VDN, the **`adjunct routing`** command operates the same as a **`route to digits`** command with coverage set to `n`. Converse: Covered calls to a VDN work with the **`converse`** command. If a call in vector processing is connected to an agent in a converse split, the agent cannot activate Consult, Coverage Callback, or Coverage Leave Word Calling. Messaging: The **`messaging`** command handles covered calls differently depending on whether an extension is specified in the command. If the command **`messaging split`** *xxxx* extension `none` is used, the mailbox of the principal extension is used for the call. The number of the principal extension and the reason for redirection are passed to the messaging adjunct in the CONNECT message. When an extension is specified in the **`messaging`** command, no information on the principal extension is passed to the adjunct. Instead, the number of the extension specified in the command is passed to the adjunct in the CONNECT message along with the reason for redirection. The mailbox for the specified extension is used. Route-to: A call covering to a VDN can be routed to any valid destination by the **`route-to`** command. When the **`route-to`** command terminates a covered call locally, information identifying the principal and the reason for redirection are retained with the call. This information can be displayed on display phones or passed to an AUDIX or a Message Center system.
Class of Restriction	The COR assigned to the covering VDN governs the vector routing of the call.
Conference	Calls in an established conference do not cover to a VDN. Once a call covers to a VDN, a conference cannot be established until the call is delivered to an extension and vector processing ends.
Consult	The feature normally uses a Temporary Bridged Appearance on the principal's set. Call coverage to a VDN removes the Temporary Bridged Appearance from the principal's set, but the Consult feature still works.

Table continues…

Interaction	Description
Hunt Groups	A VDN can be the last point in a hunt group's coverage path. If the coverage vector for a split or hunt group routes calls to another using a **`route-to`** or **`messaging`** command, calls queue at the second resource with the queue priority assigned for the first split or hunt group. If a **`queue-to`**, **`check`**, or **`converse`** command is used, calls queue at the second split or hunt group with the priority specified in the command. If an inflow threshold has been assigned to a hunt group, the group does not allow new calls to queue when the oldest call in queue has exceeded the threshold. Therefore, covered calls are not connected to a hunt group when the group's inflow threshold has been exceeded. The interaction can also occur when a **`messaging split`**, or **`route-to`** command routes a covered call to a split that is not vector-controlled.
Look-Ahead Interflow	For calls that have covered to a VDN, LAI works like a **`route-to digits/number`** vector command with coverage = n. Any Dialed Number Identification Service (DNIS) digits sent with the interflowed call indicates the VDN to which the call covered, not any VDN the call encountered before it went to coverage.
Night Service	Calls that have covered to a VDN cannot be redirected by Night Service.
Personal CO lines (PCOL)	A VDN can be assigned as the last point in a PCOL coverage path.
Phone Display	Calls covering to a VDN and then directed to an agent in a split/hunt group by a **`queue-to`**, **`check`**, **`converse`**, or **`route-to`** command display the following information to the agent. `a=EXT 3174   to   EXT 3077     b` In this example, station A calls station B. Station B is busy, and the call covers to a VDN.
Redirection on No Answer (RONA)	RONA applies to calls that cover to a VDN. If the vector associated with the VDN queues the call to a resource (for example, a split or agent) that uses RONA, the call can be requeued for the same resource. The call cannot be redirected, however, since it has already covered to the VDN unless RONA occurs as a result of a **`route-to`** command with cov=y vector step and the **Cvg Enabled for VDN Route-To Party** field applies.
Terminating Extension Groups	A VDN can be assigned as the last point in the coverage path for a Terminating Extension Group (TEG), since it has already covered to the VDN.
Transfer	Calls can be transferred to extensions that cover to a VDN. Users who receive a covered call can transfer the call to a VDN. If a transfer attempt goes to coverage and covers to a VDN, the user at the answering station can complete the transfer by pushing the Transfer button or by flashing the switchhook on an analog station. Calls that cover to a VDN can be subsequently transferred to AUDIX. Calls can also be transferred out of AUDIX to a VDN.

VDN Override

The following table depicts how the *active* VDN extension is replaced when a call is routed through a series of VDNs by the **`route-to number`** or **`route-to digits`** vector steps.

The *active* VDN extension is determined by enabling the **Allow VDN Override** field for one of the previous VDNs to which the call was routed using a **`route-to`** command based on the following rules:

- If the **Allow VDN Override** field in the previous VDN is set to `y`, the extension of the current VDN overrides the active VDN extension.
- If the **Allow VDN Override** field in the previous VDN is set to `n`, the current active VDN extension remains the same.

The following example describes the VDN Override control of the active VDN extension for all calls routed to multiple VDNs by vector processing. VDN 1 is always the initial active VDN for the call.

Settings assigned for the Allow VDN Override field on the VDN screen								
VDN 1	y	n	n	n	y	y	y	n
VDN 2	y	y	n	n	n	n	y	y
VDN 3	y	y	y	n	y	n	n	n

Active VDN after the call is routed to the next VDN in the sequence								
After call is routed to VDN2	VDN2	VDN1	VDN1	VDN1	VDN2	VDN2	VDN2	VDN1
After call is routed to VDN3	VDN3	VDN3	VDN1	VDN1	VDN2	VDN2	VDN3	VDN3

Note:

With Expert Agent Selection (EAS) enabled for the system, if the **Allow VDN Override** field is set to `y` for the original VDN, the VDN skills (defined on page 1 of the Vector Directory Number screen) of the new VDN are used for vector commands where the skill group can be administered as 1st, 2nd, or 3rd. If the **Allow VDN Override** field is set to `n` on the original VDN, the VDN skills of the original VDN are used for such vector commands.

For Best Service Routing (BSR), if the **Allow VDN Override** field is set to `y` for the original VDN, the settings for the BSR application and **Available Agent Strategy** field (defined on page 2 of the Vector Directory Number screen) of the new VDN are used for BSR-related vector processing. If the **Allow VDN Override** field is set to `n` for the original VDN, the settings for the BSR application and **Available Agent Strategy** field settings of the original VDN are used for BSR-related vector processing.

VDN parameters associated with the active VDN

With VDN Override, you can use the information on the subsequent VDN to which a call is routed to be used instead of the information on the previously active VDN. The new VDN then becomes the *active* VDN for the call. The information that is associated with the *active* VDN and with the call as the call progresses through the vector steps includes the following:

- VDN Name
- Tenant Number (TN)
- VDN of Origin Announcement (VOA) Extension
- VDN Skills (1st, 2nd, 3rd)
- VDN Return Destination
- VDN Timed After Call Work (ACW) Interval
- Best Service Routing (BSR) Application
- BSR Available Strategy
- BSR Tie Strategy
- Display VDN for Route-to Direct Agent Call (DAC)
- Trunk Adjunct Switch Application Interface (ASAI) Messenger
- BSR Local Treatment
- VDN Variables
- VDN Time Zone Offset
- Apply Ringback for Auto-Answer Calls

Application of VDN Override

You can use VDN Override in conjunction with a vector that prompts the caller for a service type. For example, a call is placed to an automobile dealer with several departments, including Sales and Parts. If the caller wants to talk to someone in Sales, the VDN named Main directs the call to the main vector which redirects the call to the Sales vector. If VDN Override is assigned to the Main VDN, the agent phone displays the Sales VDN name when the call is connected to the agent.

Note:

When you enable **Variables in Vectors**, the VDN Override settings change the VDN extension number that is assigned to a *vdn* type vector variable. VDN Override is based on the *active* VDN for the call.

VDN Override for ASAI messages

You can use VDN Override when:

- A Computer Telephony Integration (CTI) application has set up an ASAI VDN or station event-notification association.
- The CTI application requires the called number ASAI message information to be the *active* VDN extension associated with the incoming trunk or internal call rather than the called number digits contained in the ISDN SETUP message for the incoming call.

VDN Override is useful for a CTI application that is monitoring a call where the *active* VDN extension is changed by the following vector scenario:

1. An incoming call is routed to a VDN. The vector prompts the caller to enter more than one digit.
2. The call is then routed to a subsequent VDN by a **`route-to number`** or **`route-to digits`** vector step.

ASAI messages

The VDN Override for Adjunct-Switch Application Interface (ASAI) Messages feature affects the following *called number* ASAI message information:

- Call Offered
- Alerting
- Queued
- Connect
- Adjunct Route-Request

Important:

Communication Manager activates this feature for an incoming ISDN trunk call when Communication Manager routes the call to a VDN that has the **VDN Override for ASAI Messages** field on the Vector Directory Number screen set to `ISDN Trunk`. To use the VDN Override for ASAI Messages feature for ISDN trunk calls and internal calls, set the field to `all`.

When Communication Manager activates this feature for a call, this feature remains in effect for the call regardless of the **VDN Override for ASAI Messages** field setting for subsequent VDNs to which the call is routed.

The **VDN Override for ASAI Messages** field setting affects the called number information in the following manner:

- If you set the field to `no`, the called number information is the called VDN extension in the called number Information Element (IE) sent in the incoming ISDN SETUP message or the local call number and does not change after routing to the called VDN and to subsequent VDNs.

- If you set the field to `ISDN Trunk`, when an incoming ISDN trunk call is routed to this VDN, the called number information sent in the ASAI event and Adjunct Route-Request ASAI message is the *active* VDN extension that becomes associated with the call based on the VDN Override rules. This field option does not apply to internal calls.
- If you set the field to `all`, the *active* VDN is used for the called number for all types of calls to the VDN including internal calls and incoming ISDN trunk calls.

VDN Override interactions

- If you set **VDN Override for ASAI Messages** to `ISDN Trunk` or `all`, Communication Manager does not keep the field setting when an ACD agent or a station user answers an incoming call. The Communication Manager call-transfer or station call-conference features transfer or conference the call with another agent or station user.
- If Communication Manager routes an incoming Central Office (CO) call to a VDN that has the **VDN Override for ASAI Messages** field set to `ISDN Trunk` or `all`, the field setting has no effect on the called number information for the ASAI messages.
- For calls that are routed through VDNs and are answered by Call-pickup button, then VDN override flag does not get applied.

VDN Return Destination

VDN Return Destination (VRD) allows calls to be placed back in vector processing after all parties, except the originator, drop the call. The originator can optionally be an internal call or connection as well as an external trunk. Internal calls include any type of calls that can be placed to, covered to, or transferred to a VDN that is not treated as an incoming trunk call.

The **Return Destination** field on page 2 of the Vector Directory Number screen allows you to enter a VDN extension as a return destination. The VDN which has this field administered is called the VDN with this feature active. The return destination VDN (the one specified in the field) is referred to as the *Return Destination*. Return destination is applied on the VDN that is treated as active for a call when the call left vector processing.

Every call that is processed through a VDN with this feature active is routed back to the assigned VDN when all parties on the call, except the originator, drop. For this feature, the originator is the party that originated the call at the time the call entered the VDN with this feature active.

Note:

VRD does not apply to DCS and attendant handled calls.

The VDN to which the call is placed to (when the originator is the only remaining party) is determined by the return destination. The VDN can be the original VDN or a different one.

The VRD status comes from the active VDN for the call when it finally reaches the destination. Once the call has been through vector processing, the VRD status cannot be changed by subsequent vector processing. The VRD status is that it either does not have a VRD that applies (not the right origin or the active VDN assignment is blank) or that it has a VRD (an assigned VDN) that applies. If an internal call is released prior to the caller dropping, the assigned VRD extension and **Call Origin** value on the active Vector Directory Number screen determines VRD eligibility. An internal call is eligible for VRD if there is a VRD destination assigned and the **Call Origin** field on the Vector Directory Number screen is set to `internal` or `both`. VRD internal calls include:

- Station or Voice Response Unit (VRU)/Interactive Voice Response (IVR)/Voice Portal (VP) adjunct to a VDN
- An agent call to a VDN
- Agent/station/line port transfer of a previously ineligible call to a VDN
- Coverage or forwarding of a previously ineligible call to a VDN
- ASAI 3rd Party Makecalls to a VDN
- ASAI Merge connections, including Single Step Conference (SSC) to a VDN

Note:

An attendant call to a VDN or a call handled by an attendant is not eligible for VRD. If the call with a VRD reaches the PBX attendant, the VRD status will be canceled.

VDN Override applies to internal calls during the initial vector processing (not after leaving vector processing) if the call is routed to another VDN via a **`route-to`** command or adjunct route in the same manner as external calls. In this case the call origin criteria is also overridden so if for example an internal call to VDN A has **call origin** set to `internal` and override set to `y`, routing to VDN B, which has origin set to `external`, will remove the VRD. If VDN B does not have a VRD defined, the VRD will be also removed from the call since it overrides the assignment to VDN A. This feature keeps the call active after the original call has terminated. One typical use of VRD is for returning the caller, after being handled by an agent, to a VDN that is assigned to vector that routes to a VRU that surveys the caller. Another example of an application using VRD is to give the caller an opportunity to signal the need for sequence dialing (by entering a #). There are two ways this can happen:

1. When the destination drops on its own (after having answered), the call goes to the return destination which has a **`collect digits`** vector step. This step tries to collect the # sign entered by the caller.
2. When the call is not answered, the caller enters the # to request sequence calling (the ASAI-Requested Digit Collection feature collects the # sign). This # is reported to the adjunct. The adjunct requests the third_party_drop (or third_party_end_call) for the destination, and at that point the call goes to the return destination.

The VRD and ASAI-Requested Digit Collection features can be used independently, with the following rules:

1. If there is no ASAI request to collect digits, but a return destination is provided: when all parties, except the originator, drop, the switch routes the call with only one party active (the caller) to the return destination. At this point, the call enters vector processing for the VDN specified by the return destination.

The caller keeps returning to the same return destination indefinitely until either the caller hangs up or a busy or disconnect vector step is executed. Once a call leaves vector processing for the first time, the return destination is not changed.

2. If a request is made to collect digits but there is no return destination provided, the switch collects and passes the digits to an adjunct, which takes an action. However, if the adjunct drops one party on the call, the switch will drop the other party as well and clear the call (it cannot retain a call with only one party, if there is no return destination for further processing).

VDN variables

With VDN variables, you can:

- Assign up to nine variable fields, V1 through V9, on the VDN screen.
- Use the VDN variables in all vector commands that support variables except as a for parameter with the `collect-digits` command.
- Use as an operand to the `set` command.
- Use up to 16 digits to assign a number to the VDN variable and use up to 15 characters to describe the VDN variable.
- Use VDN variables as indirect references to announcement extensions and other numerical values in vector commands.
- The VDN variables assigned to the active VDN for the call are used in processing the vector.

You can create general purpose vectors that support multiple applications with call wait treatments customized for your application.

Call centers have many vectors that use the same basic call flow, but are unique because each requires unique announcements, route-to destinations, holiday tables, Vector Routing Table (VRT) indices, and conditional limits. With VDN variables, you can create a generic call flow vector. The unique items are designated on the VDN screen using VDN variables. VDN variables reduce the number of vectors, ensure common flows and ease of administration when the flows need to change due to unforeseen events such as problems with trunking, staffing, or messaging.

VDN variable fields

Each VDN variable field has a maximum 15-character description and 16-digit assignment as described in the following table.

Variable	Description	Assignment
V1	ABCDEFGHIJKLMNO	1234567890123456
V2... V9	ABCDEFGHIJKLMNO	1234567890123456

Use the **Description** field to describe the VDN variable using up to 15 characters.

Use the **Assignment** field to assign up to 16 digits to the VDN variable. Each digit entry can be:

- `0 - 9`
- `blank`

Using VDN variables with vector commands

You can use the VDN variables in all vector commands that support vector variables except as a for parameter with the **`collect digits`** command.

Announcement command

You can enter a VDN variable between V1 - V9 as an announcement extension in all commands that use an announcement in the extension field.

The following syntax rules apply when VDN variables are used with the **`announcement`** command.

```
announcement [V1-V9]
collect [1-16] digits after announcement [V1-V9]
for [none, A-Z, AA-ZZ]
disconnect after announcement [V1-V9]
wait-time [0-999 secs, 0-480 mins, 0-8 hrs] hearing [V1-V9] then [music, ringback,
silence, continue]
```

Requirements and considerations

- You can use a VDN variable or a vector variable, but not both.
- When the command is executed, the assignment entry for that variable is taken from the VDN screen for the active VDN of the call and used as the announcement extension number.
- The number must be a valid announcement extension assigned on the Audio/Announcement screen.

Converse-on command

The following syntax rules apply when VDN variables are used with the **`converse-on`** command.

```
converse-on skill [hunt group, 1st, 2nd, 3rd]
pri [l, m, h, t] passing [data1] and [data2]
converse-on split [hunt group] pri [l, m, h, t] passing [V1-V9] and [V1-V9]
```

Note:

You can use a VDN variable in data1, data2, or in both.

Disconnect command

You can use VDN variables with the **`disconnect`** command after an announcement extension.

The following syntax rules apply when using VDN variables with the **`disconnect`** command.

```
disconnect after announcement [V1-V9]
```

Goto commands

The following syntax rules apply when using VDN variables with the **`goto`** command.

`goto` **step 1-99 if** **or** `goto` **vector 1-2000 @ step1-99 if**						
A-Z, AA-ZZ	>,<, =,<>, >=, <=	*V1-V9*				
	in table	*V1-V9*				
	not in table					
ani	>, >=, <>, =, <, <=	*V1-V9*				
	in table	*V1-V9*				
	not in table					
available-agents	in skill	hunt group, skills for VDN: 1st, 2nd, 3rd	<, >=, <>, =, <, <=	*V1-V9*		
calls-queued	in skill	hunt group, skills for VDN: 1st, 2nd, 3rd	pri	priorities: l = low, m = medium, h = high, t = top	>, >=, <>, =, <, <=	*V1-V9*
counted-calls	to vdn	vdn extension, latest, active	>, >=,<>, =, <, <=		*V1-V9*	
digits	>, >=,<>, =, <, <=	*V1-V9*				
	in table	*V1-V9*				
	not in table					
expected-wait for	best	>, >=, <>, =, <, <=	*V1-V9*			
	for call					
	for split	hunt group	pri	priorities: l = low, m = medium, h = high, t = top	>, >=, <., =, <,<=	*V1-V9*
	for skill	hunt group, skills for VDN: 1st, 2nd, 3rd				
holiday	in table	*V1-V9*				
	not in table					
ii-digits	>, >=, <>, =, <, ,=	*V1-V9*				
	in table	*V1-V9*				
	not in table					

Table continues…

`goto` step 1-99 if **or** **`goto` vector 1-2000 @ step1-99 if**						
interflow-qpos	>, >=, <>, =, <, <=	*V1-V9*				
oldest-call-wait	in skill	hunt group, skills for VDN: 1st, 2nd, 3rd	pri	priorities: l = low, m = medium, h = high, t = top	>, >=, <>, =, <=	*V1-V9*
rolling-asa	for skill	hunt group, skills for VDN: 1st, 2nd, 3rd	>, >=, <>, =, <,<=	*V1-V9*		
staffed-agents	in skill	hunt group, skills for VDN: 1st, 2nd, 3rd	>, >=, <>, =, <, <=	*V1-V9*		
V1-V9	>, <, =, <., >=, <=	*V1-V9*				
	in table	*V1-V9*				
	not in table					
wait-improved for	best	>, >=, <>, =, <, <=	V1-V9			
wait- improved for	best	>, >=, <>, =, <, <=				*V1-V9*
	skill	hunt group, skills for VDN: 1st, 2nd, 3rd	pri	priorities: l = low, m = medium, h = high, t = top	>, >=, <>, =, <=	
	split	hunt group				

Route-to command with VDN variables

The following syntax rules apply when VDN variables are used with **`route-to`** number commands.

`route-to`	number	*V1-V9* ~r [*V1-V9*] (~r indicates that Network Call Redirection is attempted.)	with cov	`y` `n`	if	digit	>, >=, <>, =, <=	0-9, #
						interflow-qpos	<, =, <=	1-9
						unconditionally		

Requirements and considerations

- You can use a variable in the number field as the destination address for the **`route-to`** command. When the route-to number [*V1-V9*] step is executed, the current numerical value of up to 16 digits is used as the destination address.
- If the variable is not defined, the route-to step fails and a “variable not defined” vector event 38 is logged. Vector processing continues at the next vector step. The destination number retrieved from the string of digits of the current value of the variable must be a valid destination as defined by the Communication Manager dial plan. Otherwise, the **`route-to`** command fails to log the appropriate vector event and vector processing continues at the next step.

Set command with VDN variables

The following syntax rules apply when using VDN variables with the **`set`** command.

```
set [variables, digits] = [operand1] [operator] [operand2]
```

You can use VDN variables in operand 1 and 2.

Wait command with VDN variables

The following syntax rules apply when VDN variables are used with the **`wait`** command.

```
wait-time [0-999 secs, 0-480 mins, 0-8 hrs]
hearing [V1-V9] then [music, ringback, silence, continue]
```

Case studies

Using one vector for different announcements

In this case study, agents working for the Alpha service bureau handle calls for three different companies - ABC, XYZ , and JYK. The processing for all three companies is the same, but the announcements are different.

Since the processing is identical, Alpha decides to use the same vector for all three call types. VDN variables make this possible because Alpha can use a VDN variable to define the different announcement extensions. Each call type is routed to three different VDNs - one for each company. In this example, the V1 VDN variable defines the different announcement extensions used for the initial announcement in the vector. All three VDNs are assigned to vector 5.

VDN	VDN description	V1 VDN variable assignment	Announcement
1000	ABC Company	3000	*You have reached the ABC Company …*
1001	XYZ Company	3001	*You have reached the XYZ Company …*
1002	JYK Company	3002	*You have reached the JYK Company …*

Vector 5

```
1. wait-time 0 secs hearing ringback
2. queue-to skill 10 pri l
3. announcement V1
4. wait-time 60 secs hearing music
5. announcement 3010 [All our agents are still busy.]
6. …
```

Business case

In the case study, the XYZ company has a separate vector for each application. Using VDN variables, the company can consolidate similar vectors that are each reached by a different VDN, into one vector. The company plans to use the newly-freed vectors for other applications. The problem is that the number of different parameters or values required to be assigned to the VDNs as VDN variables exceeds the limit of five variables.

The case study shows a method for combining parameter values into digit strings of up to 16 digits. Each digit string can be assigned to the VDN variables, separated into the component parts and assigned to vector variables in the common vector for each of the vector commands.

Current configuration

Before vector consolidation, all vectors had the same basic structure as shown in vector 1 for calls to VDN 1. In spite of this similarity, each vector has the following differences:

- Three different extension numbers for the announcements
- Two different Vector Routing Tables for digit checking
- Three different route-to number destinations
- A different messaging skill mailbox extension
- A different skill for queuing the call and for the messaging skill. These can be assigned using the skill preferences fields on the VDN screen.

```
Vector 1
1. wait-time 0 secs hearing ringback
2. collect 4 digits after announcement 1001 for none
3. goto vector 300 @step 1 if digits in table 11
4. goto vector 301 @step 1 if ani in table 12
5. goto step 13 if expected-wait for skill 100 pri l > 600
6. queue-to skill 100 pri l
7. announcement 1002
8. wait-time 120 secs hearing 1003 then music
9. route-to number 2001  [LAI looking for an available agent at location 1]
10. route-to number 2002 [LAI looking for an available agent at location 2]
11. route-to number 2003 [LAI looking for an available agent at location 3]
12. goto step 7 unconditionally
13. messaging skill 210 for extension 5001
```

How to assign parameters

Following are the parameters assigned for three VDNs. The parameters appear in the vector in the same order as described in the following table.

Parameter	VDN 1	VDN 2	VDN 3
announcement extension 1 for collect step	1001	1010	1100
VR table 1 for digits	11	21	31
VR table 2 for ani	12	22	32
queuing skill (first)	100	200	300
announcement 2	1002	1012	1102
audio source 3 for wait command	1003	1013	1103
route-to destination 1	2001	3001	4001
route-to destination 2	2002	3002	4002
route-to destination 3	2003	3003	4003
messaging skill hunt group (second)	210	310	410
messaging mailbox extension	5001	5002	5003

How to group parameters

One way to combine parameters is to group the parameters by function. For example, combine all announcements into one VDN variable. The following table describes this approach.

VDN variable	Parameter	VDN 1	VDN 2	VDN 3
V1	announcement extension 1 for collect step	1001	1010	1100
	announcement 2	1002	1012	1102
	audio source 3 for wait command	1003	1013	1103
V2	VR table 1 for digits	11	21	31
	VR table 2 for ani	12	22	32
V3	route-to destination 1	2001	3001	4001
	route-to destination 2	2002	3002	4002
	route-to destination 3	2003	3003	4003
V4	messaging mailbox extension	5001	5002	5003
Skill Preferences	queuing skill (first)	100	200	300
	messaging skill hunt group (2nd)	210	310	410

How to assign digit strings

The string of digits to be assigned to each VDN variable on the VDN screen is described in the following table. The order is based on how the subroutine is written to separate the components. The capital letters A through H reference the vector variables that are used in the common processing vector.

VDN variable	Description	VDN 1	VDN 2	VDN 3
V1	Three announcements: A, B, C	100110021003	101010121013	110011021103
V2	Two table values: D, E	1112	2122	3132
V3	Three route-to destinations: F, G, H	200120022003	300130023003	400140024003
V4	mailbox	5001	5002	5003
Skill Preferences	1st	100	200	300
	2nd	210	310	410

Note that VDN variables V5 through V9 are not used in this example.

How to separate and assign parameters to vector variables

Vector 1 is the common vector for incoming calls that go to VDN 1, VDN 2, and VDN 3. Vector 1 is modified to include a subroutine call to vector 2 that separates the combined parameters assigned to each VDN variable and assigns the parameters to the correct vector variables in vector 1.

```
Vector 1 - revised to serve as the common vector for calls to VDN1, VDN2 and VDN3
1. wait-time 0 secs hearing ringback
2. goto vector 2 @step 1 if unconditionally
3. collect 4 digits after announcement A for none
4. goto vector 300 @step 1 if digits in table D
5. goto vector 301 @step 1 if ani in table E
6. goto step 14 if expected-wait for skill 1st pri l > 600
7. queue-to skill 1st pri l
8. announcement B
9. wait-time 120 secs hearing C then music
10. route-to number F  [LAI looking for an available agent at location 1]
11. route-to number G [LAI looking for an available agent at location 2]
12. route-to number H [LAI looking for an available agent at location 3]
13. goto step 7 if unconditionally
14. messaging skill 2nd for extension V4
```

How to define vector variables

The A through H vector variables need to be defined on the Variables for Vectors screen as the *collect* type with local scope as described in the following table. The **Assignment** and **VAC** fields are left blank.

Var	Description	Type	Scope	Length	Start
A	announcement 1	collect	local	4	1
B	announcement 2	collect	local	4	1
C	announcement 3	collect	local	4	1
D	table 1 (digits)	collect	local	2	1
E	table 2 (ani)	collect	local	2	1
F	route to 1	collect	local	4	1
G	route to 2	collect	local	4	1
H	route to 3	collect	local	4	1

How to separate each VDN variable

Vector 2 is the subroutine vector and separates each VDN variable into component parts.

```
Vector 2
1. set A = V1 SEL 12 [A = 1001 when V1 = 100110021003 since A being of length 4 is
assigned only the leftmost 4 digits]
2. set B = V1 SEL 8 [B = 1002 since SEL selects 10021003 and B being of length 4 is
assigned only the leftmost 4 digits]
3. set C = V1 SEL 4 [C = 1003 since SEL selects the rightmost 4 digits]
4. set D = V2 SEL 4 [D = 11 when V2 = 1112 since D being of length 2 is assigned only
the leftmost 2 digits]
5. set E = V2 SEL 2 [E = 12 since SEL selects the rightmost 2 digits]
6. set F = V3 SEL 12 [this step and following functions the same as for A, B, and C]
7. set G = V3 SEL 8
8. set H = V3 SEL 4
```

Summary

The case study described how to use the VDN variables to support eight parameters. The case study also described how to use variable V4 for another parameter that required to be passed with the active VDN for the call. The approach supported nine parameters with four VDN variables while keeping V5 as a spare. Since *A-H* vector variables are local variables, you can reuse the variables in other vector applications with similar string lengths.

Vector subroutines

Subroutines use common vector programs that can be used by different vectors without duplicating the same sequence in each vector. Subroutines can significantly decrease the number of steps and vectors required.

The goto step is used for vector subroutines. The goto step uses:

- The @ step parameter to branch to a specific step in the vector
- The **`return`** command to return from a subroutine

The maximum simultaneous active subroutine calls allowed are:

- 8000 for Avaya S8500, S87XX, S88XX, and Avaya Common Server platforms
- 400 for Avaya S8300

With vector subroutines, you can reuse common sets of vector commands. For example, you can use a single subroutine for all vectors to determine if a call has arrived within business hours. Without a subroutine, each vector repeats the step.

Following are some of the advantages of using vector subroutines:

- More steps per vector by removing duplication.
- Unused steps at the end of vectors can be used for subroutines, expanding vector capacity.
- Ease of administration. You can change just one vector subroutine that is referenced by many vectors, such as changing office hours or wait treatment.

The goto command and subroutines

Use the **goto vector** command to branch vector processing to a subroutine or to a specific step in the vector. The **goto vector** command works with the **return** command to return vector processing to the calling vector. When a **goto vector** command is executed, the vector and the subsequent step number for the command are stored with the call. This is the return destination that is used with subroutines.

When the **goto vector** command branches to the specified vector, any data associated with the call remains with the call. Examples of call-associated data are collected digits and dial-ahead digits. Changes made stay with the call when the call returns to the calling vector.

The @ step parameter

Use the @ step parameter with the **goto vector** command conditionals to branch to a specific step in a vector. For example:

- **goto vector xxx @ step yy if <conditional> [comparator] <threshold>**
- **goto vector xxx @ step yy if unconditional**

The requirements for the @ step parameter are as follows:

- The default step number is 1. The step number remains at 1 until you change it to a step number between 2 and 99.
- When the step number is set between 1 and 99, the **goto vector** command saves the returned data when subroutines are active. Vector processing starts again at the branched-to vector at the specified step.
- If the specified step in the branched-to vector is blank, vector processing skips to the next step in the vector. If it is the last step, it is treated as a stop step.

Example 1: Test for working hours

The call center of the XYZ retail stores has a large number of vectors to check if calls arrive during working hours or not. Before the availability of vector subroutines, each vector required the same series of steps to test for working hours. With vector subroutines, only one vector is required for the series of steps that check for working hours. Each vector that requires the check uses a **goto vector** step to run the tests. Vector processing returns to the step following the calling **goto vector** step if the test passes. Otherwise, the *out-of-working* hours treatment is given by the subroutine.

The call center edits just one vector and the change is reflected in other vectors that reference this vector.

Incoming call processing vector example

```
1. wait 0 secs hearing ringback
2. goto vector 20 @step 1 if unconditionally
3. queue-to skill 100 pri l [subroutine returns here if call is during working hours]
4. announcement 1000 [
```

```
Thank you for calling XYZ Retail Stores, your call is important to us]
5. …
```

Checking working hours vector subroutine example

```
Vector 20
 1. goto step 9 if time-of-day is all 23:00 to all 07:00
 2. goto step 9 if time-of-day is Friday 21:00 to sat 07:00
 3. goto step 9 if time-of-day is sat 16:00 to sun 07:00
 4. goto step 9 if time-of-day is sun 16:00 to mon 07:00
 5. goto step 7 if holiday in table 5
 6. return [call is during working hours]
 7. announcement 2001 [The XYZ Stores are closed on holidays.]
 8. goto to step 10 if unconditionally
 9. announcement 2001 [You have called after the XYZ Stores are closed.]
10. disconnect after announcement 2001 [Please call back during normal business hours:
7 am to 11 pm on
Monday through Thursday, 7 am to 9 pm on Friday and 7 am to 4 pm on Saturday and
Sunday.]
```

Voice Response Integration

VRI

Voice Response Integration (VRI) integrates Call Vectoring with the capabilities of Voice Response Units (VRUs).

You can perform the following tasks:

- Pass data between the system and a VRU.
- Pool IVR ports for multiple applications.
- Run a VRU script while a call is in a queue, retaining the position of the call in the queue.
- Run a VRU script while retaining control of a call in vector processing.
- Tandem VRU data through a communication server to an ASAI host.
- Use a VRU as an external announcement device.
- Provide agent identifier in a system variable for vectoring use, when a customer call is redirected by the VDN Return Destination (VRD) feature into vector processing.

The **converse-on** command, which is part of Basic Call Vectoring, provides these capabilities. Use a converse-on step to integrate a VRU with ACD. With VRI, you can use the VRU capabilities while controlling a call in ACD.

Include VRUs with vector processing to take advantage of the following:

- Audiotex applications
- Caller information
- Increased recorded announcement capacity

- Interactive Voice Response (IVR) applications
- Local and host databases
- Speech recognition
- Text-to-speech capabilities
- Transaction processing applications

With VRI, contact centers can make productive use of call queuing time. For example, while a call is in a queue, the caller can listen to product information or can complete an IVR transaction. You can resolve caller queries, which reduces the time in a queue for other callers.

VRI detailed description

A call queued to a split or skill retains position in the queue while a VRU script is being run. When an agent becomes available, the line to the VRU is dropped and the caller connects to the agent.

The **`converse-on`** command delivers a call to a predetermined converse split or skill. A converse split or skill is administered and operates exactly like other splits or skills. Non converse splits or skills are accessed by the **`queue-to`** and **`check`** vector steps.

Members of a converse split or skill are the ports connected to the VRU. If all VRU ports are busy, a call queues to the converse split or skill with the administered priority. After the VRU answers the call, the **`converse-on`** command passes up to 2 data items to the VRU, depending on command parameters specified. You can pass data required by a VRU script or data that selects the VRU script to be run.

Whether or not you pass data, a caller is connected to the VRU, which runs the VRU script. Audible feedback provided by the vector is not heard and no further vector steps are run until the VRU script completes. The VRU returns data to the system and then drops the line to the system. Vector processing continues at the step following the **`converse-on`** command.

If the call was queued to a non converse split or skill before the **`converse-on`** command was run, the call retains the queue position. If an agent becomes available while the VRU script runs, the system drops the line to the VRU and connects the caller to the agent. The VRU detects the disconnect and terminates the VRU script.

With Call Prompting, you can collect and use digits that the VRU returns. The digits are handled as dial-ahead digits. Rules for collecting and processing VRU digits are the same as for Call Prompting.

You can use digits returned from the VRU in the following ways:

- To display for the answering agent, automatically for 2-line displays or with the callr-info button for other displays.
- As an extension in a **`route-to digits`** vector step. For example:

  ```
  converse-on split. . . . (VRU returns 4 digits)
  collect 4 digits after announcement none
  route-to digits coverage y
  ```

- For vector-conditional branching in an **`if digits equals`** vector step. For example:

  ```
  converse-on split . . . (VRU returns 1 digit)
  collect 1 digit after announcement none
  goto vector 101 if digits = 1
  goto vector 102 if digits = 2
  goto vector 103 if unconditionally
  ```

- Tandem to an ASAI host - collected digits are passed to ASAI hosts in call offered to domain event reports and in route request messages, therefore, caller digits or database information returned from the VRU can be sent tandem through the system to ASAI hosts. For example:

  ```
  converse-on split ... (VRU returns 9 digits)
  collect 9 digits after announcement none
  adjunct route link Y
  ```

In this vector, the digits returned from the VRU are forwarded to the ASAI host in the adjunct routing route request message.

When you use a VRU application that returns data for a **`collect-digits`** step, the opportunity for toll fraud exists when the VRU application does not return any data. Following are some guidelines:

- If the collected digits are used to route calls internally, ensure that the COR for the VDN does not allow calls to route externally.
- If the collected digits are used to route calls externally, use a password to verify that the collected digits have been passed by the VRU application. For example, in the following vector, the VRU application returns a 3-digit password followed by the 8-digit external number. The vector routes calls without the correct password to a vector 23.

  ```
  converse-on split 10 pri m passing none and none (VRU returns 11 digits)
  collect 3 digits after announcement none
  goto vector 23 if digits <> 234
  collect 8 digits after announcement none
  route-to digits with coverage n
  ```

The agent identifier is available in a system variable for vectoring use when a customer call is redirected by the VRD feature into vector processing. The agent identifier can be included in User-to-User Information (UUI) using existing vector commands before routing the call to an external IVR system. By adding the agent identifier in UUI, post call surveys can furnish reports details down to the agent level.

The *agent* type variable is read-only and cannot be set. The *agent* variable can be assigned only in the following conditions:

- By setting an *asaiuui* variable in a VRD vector
- As a **`converse-on`** operand [`operand1` or *operand2*] in a VRD vector.

The following vector example illustrates using the *agent* variable, as a parameter in the **converse-on** vector command in a VRD vector.

```
01 converse-on skill 801 pri l passing LA and none    [LA – 'agent' variable]
```

The following vector example illustrates how to set an *ASAIUUI* variable to the last agent login ID in a VRD vector. By adding the agent identifier in UUI, post call surveys can furnish reports details down to the agent level. You can set an *asaiuui* variable to the last agent login ID by adding the *asaiuui* variable to the left of the = operand and the *agent* variable to the right of the = operand, with the SEL operator and the length specified as operand2.

```
01 set CH = LA  SEL  7     [CH – 'asaiuui' variable, LA – 'agent' variable]
```

VRI interactions

Converse splits interact like other vector-controlled splits unless noted here.

Interaction	Description
Adjunct Switch Application Interface (ASAI)	When a **converse-on** vector step places a call to an ASAI-monitored domain, ASAI event messages are sent over the ASAI link. When a **converse-on** step places an ASAI-monitored call, the ALERT message sent to the ASAI adjunct includes a cause IE, Coding Standard 3 value 23 (CS3/23), which informs the adjunct that the call has not been dequeued from any non converse splits. If a **converse-on** step is run while an adjunct routing request is outstanding, the request is canceled. ASAI cannot transfer or conference calls but can direct the system to perform the action.
Agents	You can use a **converse-on** step to deliver a call to a group of human agents. To agents, the call looks like an ACD call, except that agents cannot use certain features such as Transfer, Conference, and Supervisor Assist. The agent can return data to vector processing by pushing the **transfer** button or flash hook on analog and dialing the converse-on data return code and required digits.
Answer supervision	Answer supervision is returned only once during a call. If a call is answered because of a converse-on step, answer supervision is sent if not previously sent. If digits are passed to the Voice Response Unit (VRU), answer supervision is sent after the digits are sent.

Table continues…

Interaction	Description
AUDIX	If a **converse-on** step calls AUDIX, the call is handled as a direct call to AUDIX. The caller hears the AUDIX welcome message and can retrieve messages as usual. If a call is forwarded to a VDN and then delivered to an AUDIX hunt group by a **converse-on** step, the call to AUDIX is treated as a redirected call, and the caller can leave a message.
Auto-Available Split/Skill (AAS)	A **converse-on** vector step can place a call to an AAS. Use auto-available converse splits or skills for VRI except when ASAI controls the converse split or skill.
Automatic answering	When you administer ports on the Interactive Voice Response (IVR) system as agents of a converse split or skill, do not administer agents as automatic answer. The system-provided zip tone can interfere with the interaction between the IVR system and the calling party.
BCMS or CMS	BCMS tracks calls that a **converse-on** step places to a BCMS-measured hunt group. CMS tracks calls that a **converse-on** step places to a CMS-measured hunt group, split, or skill. The VDN tracks such calls as waiting in the vector. A call is treated as answered when answered by a non converse split or skill agent, not when answered by a converse split or skill agent. The converse split or skill tracks this as a separate answered call when the VRU answers. Though trunk and split or skill totals no longer match, VDN and trunk totals match.
Call Detail Recording (CDR)	The duration of a call to a VDN is recorded from when answer supervision is returned after a successful **converse-on** step. Unsuccessful **converse-on** steps do not generate ineffective call-attempt records. **Converse-on** steps cannot place calls. The steps simply direct a call to a hunt group.
Call Park	Calls that a **converse-on** step placed cannot be parked.
Call Pickup	Do not use Call Pickup with **converse-on** steps.
Class of Restriction (COR)	The system does not check CORs when a **converse-on** vector step routes a call to a split.
Conference	You cannot conference a call routed by a **converse-on** step.
Direct Department Calling (DDC)	You can administer a converse split or skill as a DDC split or skill.
Distributed Communications System (DCS)	If an incoming DCS call is placed to a vector with a **converse-on** step, the caller DCS extension is sent to the VRU.
Expert Agent Selection (EAS)	**Converse-on** steps can place calls to a skill hunt group.
Hold	An agent answering a converse call can put the call on hold, but the caller does not hear music on hold. If a call is queued to a backup split or skill before it was sent to the VRU and a non converse split or skill agent answers the call on hold, the agent who placed the call on hold is dropped, and the caller connects to the answering agent.
Hold - Automatic	Automatic hold applies to converse-on calls.

Hunt Groups

A **converse-on** step can deliver a call to a vector-controlled or AUDIX hunt group, ACD split, agent skill, or message center.

Interaction	Description
Integrated Services Digital Network (ISDN)	You can administer a **converse-on** step to send a Calling Party Number (CPN) / Billing Number (BN) to the IVR system using the caller keyword.
Intra-switch CDR	If a converse-on call is answered and either the caller or the VDN associated with the call is administered for intra-switch recording, timing for the call is started and the CDR record shows *calling party to VDN* as the originating and answering parties.
Line-side T1 connectivity	T1 connectivity between Communication Manager and the IVR system is supported for VRI. The DS1 board must be a TN767E or later, or TN464F or later. Administer all converse agents as DS1FD-type stations. Operation of the converse step using Line-side T1 is identical to that over a tip/ring line. In particular, delay-timing and outpulsing speed is the same as for analog lines. T1 connectivity to the IVR system is supported only in the United States and Canada.
Look-Ahead Interflow (LAI)	If an incoming call or a call routed by a converse-on vector step is answered by a VRU, or is queued to the converse split or skill while an LAI call attempt is outstanding, the attempt is accepted.
Message Center	**Converse-on** steps can deliver calls to message hunt groups. Such calls are handled as direct calls to the message hunt group. If a call is forwarded to a VDN and a **converse-on** step delivers the call to a message split, the call is handled as a redirected call. A **converse-on** step can queue a call to three different skills and then to a converse skill group or split.
Music-on-Hold (MOH)	During the data return phase of a **converse-on** step, the caller is placed on hold, but does not hear music.
Non vector-controlled splits	A **converse-on** step cannot route a call to a non vector-controlled split.
Queuing	Converse-on calls queue when delivered to busy hunt groups. Call Vectoring audible feedback is not disconnected while a converse-on call is queued. If a **converse-on** step is run while a call is queued to a non-converse split or skill, the call remains in queue, even after being answered by the VRU. **Converse-on** steps can queue calls at one of four priority levels: low, medium, high or top. You administer the queue priority of a call on the **converse-on** step.

Table continues…

Interaction	Description
R2-MFC signaling	R2-MFC signaling trunks can send Automatic Number Identification (ANI) to VRUs using the *ani* data item on the **converse-on** step.
Recorded announcement	Use VRI to increase the system recorded announcement capacity by offloading some recorded announcements to a VRU. Using the **converse-on** step, redirect callers to a group of VRU ports by passing the number of the announcement to be played. The IVR system can play any announcement on any port. Although only one caller can be connected to each port, up to 48 callers can be connected simultaneously to the IVR system. The maximum number of callers that can be connected to a VRU simultaneously varies with each VRU.
Redirection on No Answer (RONA)	If a **converse-on** step calls a hunt group with *no answer time out* administered and the call rings an agent or port for longer than the time out interval, the call redirects and the agent or port is put in the AUX work mode or logged out if the agent is an AAS member. With RONA, the call is requeued to the split or skill. The call cannot requeue to the split or skill if it is an AAS with all agents logged out or if the queue is full. If the call cannot be requeued, the **converse-on** step fails, a vector event is logged, and processing restarts at the next vector step.
Service Observing	Calls delivered by a **converse-on** step can be observed. To prevent the observer from hearing tones associated with data being sent to the VRU, the observer is not connected to the call until after data is passed. If the VRU returns data, the observer is put in service-observing-pending mode and the caller is put on hold while the data is sent. When the converse-on session ends and the VRU drops the line, the observer remains in service-observing-pending mode and waits for the next call. In addition, the observer observing a VDN does not hear data being sent. After data is sent, the observer rejoins the call. Do not administer a service observing warning tone because the warning tone can interfere with the interaction between the IVR system and the caller.
System measurements	System measurements track converse-on calls to hunt groups.
Touch-Tone Dialing	A caller can use touch-tone dialing while digits are passed in a converse-on session. The data is not corrupted. The system does not collect the dialed numbers as dial-ahead digits. After the system sends digits to the IVR system, a caller can enter touch-tone digits at the IVR prompt. After the IVR system has returned data to the system and an additional collect <#> digits vector step is run, a caller can enter a touch-tone response to a system prompt.

Table continues…

Interaction	Description
Transfer	A call delivered by a **`converse-on`** step cannot be transferred. If an attempt to transfer a converse-on call is made, a vector event is logged, the line to the IVR system is dropped, and processing restarts at the next vector step. If a human agent tries to transfer a call, the transfer fails and the agent reconnects to the call.
Transfer out of AUDIX	If a **`converse-on`** step delivers a call to an AUDIX hunt group and the caller tries to transfer out of AUDIX, the transfer fails and processing continues at the next vector step.
Uniform Call Distribution (UCD)	You can administer a converse split or skill as a UCD split or skill.
VDN Display Override	If a call that accesses multiple VDNs encounters a **`converse-on`** step that passes VDN, normal display override rules determine which VDN number is sent to the VRU.
Vector-controlled splits or sSkills	**`Converse-on`** steps can deliver calls only to skills or vector-controlled splits.

Chapter 3: Call Vectoring benefits, considerations, and interactions

Call Vectoring benefits

Call Vectoring benefits	Examples
Call treatment	
Implement special treatment based on the time of day, the day of the week, and on holidays. For example, routing calls to a different vector when one location is on holiday.	Customer service center on page 256 Distributed call centers on page 260
Automatically change treatment according to either how long the call has been waiting or in response to the changing traffic or staffing conditions.	Automated attendant on page 257 Data in voice answer and data message collection on page 258 Distributed call centers on page 260 Help desk on page 262
Provide multiple or recurring information, or delay announcements that are selected according to the time of day or day of the week, call volume, or staffing conditions.	Customer service center on page 256
Set up and test special call treatments for events such as sales, advertising campaigns, holidays, snow days.	Holiday Vectoring example on page 277
Provide the caller with a menu of choices.	Data in voice answer and data message collection on page 258 Help desk on page 262
Queue calls to up to three splits simultaneously to improve the ASA and agent productivity.	Customer service center on page 256 Distributed call centers on page 260
Implement routing to local or distant destinations.	Customer service center on page 256 Data in voice answer and data message collection on page 258 Distributed call centers on page 260 Help desk on page 262

Table continues…

Call Vectoring benefits	Examples
Connect callers to a voice mail or messaging system either automatically or per caller request.	Data in voice answer and data message collection on page 258
Call routing	
Reduce call transfers by accurately routing callers to the desired destination.	Data in voice answer and data message collection on page 258
Provide up to four ACD queuing priority levels and the ability to change the queuing priority dynamically, thereby, providing faster service for selected callers.	Customer service center on page 256 Data in voice answer and data message collection on page 258 Distributed call centers on page 260
Reduce agent or attendant staffing requirements by: (1) automating some tasks, (2) reducing caller hold time, and (3) having agents in one split provide service multiple call types.	Data in voice answer and data message collection on page 258
Information collection	
Provide customized or personalized call treatment using information collection and messaging.	Automated attendant on page 257 Data in voice answer and data message collection on page 258 Help desk on page 262
Collect information for use by an adjunct or by agent display.	Help desk on page 262
Collect caller-entered or customer database-provided CINFO digits from the network.	CINFO vector example on page 42
Provide an *agent* variable that maintains the “Last EAS Agent” to drop off the call throughout the life cycle of the call. The *agent* variable can be passed in the UUI to Experience Portal or other SIP-connected adjuncts, which you can then tailor to include in your scripts or store the agent identifier with call survey data.	Example application using a vector variable in other commands on page 215 Example application using a vector variable in the converse-on command on page 215

Call Vectoring considerations

Displaying VDN names for vector-initiated DACs

The Display VDN for Route-to DAC feature improves the efficiency of call center agents who answer vector-initiated direct agent calls that originate from multiple Vector Directory Numbers (VDNs).

The type of information displayed at the agent station display with a vector-initiated direct agent call can be summarized as follows:

- When the Display VDN for Route-to DAC feature is not enabled, only the EAS LoginID name for the agent who receives the call is shown.
- When the Display VDN for Route-to DAC feature is enabled for such calls, the active VDN name associated with the call is shown.

Providing agents with the ability to see the VDN name associated with an incoming call improves agent efficiency and customer satisfaction. For example, if an agent receives incoming trunk calls for different products from three different VDNs, the VDN name displayed by the Display VDN for Route-to DAC feature allows the agent to answer the call as a sales representative of that product. This feature is especially useful when vector-initiated direct agent calls route incoming trunk callers to personalized agent providing services for new customers, special product offers, or premier levels of service.

Display VDN for Route-to DAC feature operation

The Display VDN for Route-to DAC feature is designed for call scenarios where a VDN-initiated call is routed to a vector where direct agent calls are originated by one of the following methods:

- A route-to number vector step with cov parameter set to y, where the **number** field is administered with a valid EAS loginID extension.
- A route-to digits vector step with coverage parameter set to `y`, where a **`collect digits`** vector step preceding this step is used to allow the caller to enter the digits for an EAS LoginID extension.
- An adjunct routing link vector step, where a direct agent call is originated by the Route Select digit information returned from a CTI application.

The Display VDN for Route-to DAC feature is activated for an incoming trunk call when the call is routed through a VDN that has the **Display VDN for DAC Calls?** field administered to `y`. When one of the above-listed vector steps routes such an incoming call as a direct agent call to an EAS loginID extension, the active VDN name is shown on the called agent station display instead of the called EAS agent's LoginID name. If this call is routed to another EAS agent in the initially-called EAS agent coverage path, the active VDN name will again be shown on the covered-to agent station display, instead of the initially-called EAS agent LoginID name.

Station display formats

If the Display VDN for Route-to DAC feature is activated for an incoming trunk call routed through a VDN to a vector that initiates a direct agent call to an EAS agent, the format of the called agent station display appears as one of the following:

```
<Incoming Trunk Name> to <VDN Name>
```

```
<Incoming caller ANI> to <VDN Name>
```

If the Display VDN for Route-To DAC feature is not activated for an incoming trunk call, the called agent station display appears as one of the following:

```
<Incoming Trunk Name> to <EAS loginID extension>
```

```
<Incoming caller ANI> to <EAS loginID extension>
```

Note:

If the EAS agent to which the call is routed by vector-initiated Direct Agent Calling (DAC) is not available, and the called EAS agent has a coverage path to other EAS agents, the Display VDN for Route-to DAC feature preserves the active VDN name and sends it to the agent station display for a covered-to EAS agent. If the call covers to a normal station extension in the called EAS agent coverage path, the Display VDN for Route-to DAC feature does not apply to the covered-to station display, and the EAS LoginID of the called EAS agent is displayed instead.

Methods for creating vectors that use the Display VDN for Route-to DAC feature

You can administer a vector in several different ways to utilize the Display VDN for Route-to DAC feature.

Note:

For any of the vector examples shown below, if an incoming trunk call is routed through a **VDN with the Display VDN for Route-to DAC?** field set to `y`, the direct agent call is activated with the VDN Display for Route-to DAC feature.

Using collect digits and route-to digits commands

The following vector example shows how to:

- Use a collect digits vector step to prompt a caller to enter digits for a valid EAS agent loginID extension
- Use a route-to digits vector step to route the call to an agent as a direct agent call:

```
wait-time 0 secs hearing ringback
collect 5 digits after announcement 3001
go to step 5 if digits < > 1????
route-to digits with coverage y
announcement 3002
goto step 2
```

Using route-to number commands

The following simple vector uses the route-to number vector step to originate a direct agent call to an EAS LoginID extension:

```
wait-time 0 secs hearing ringback
route-to number 85103 with cov y
```

Using adjunct routing link commands

You can also originate a direct agent call with a vector that includes an adjunct route vector step. When an incoming trunk call is routed through a VDN to a vector that includes an adjunct route vector step, vector processing treats this step like a **route-to number with cov** set to y vector step.

The following vector uses the adjunct route vector step to originate a direct agent call. In this example, the CTI application is designed to route the call as a direct agent call in a Route Select ASAI message.

```
1. wait 0 secs hearing ringback
2. adjunct route link 3
3. wait 30 secs hearing ringback
4. announcement 3501
5. disconnect
```

Interactions with other Communication Manager features

Interactions of the Display VDN for Route-to DAC feature with other Communication Manager features include the following:

Interaction	Description
Call Coverage	When the Display VDN for Route-to DAC feature is activated for a call, and a vector-initiated direct agent call is made to an EAS agent having a coverage path that has other agents as coverage points, the active VDN name associated with the call is displayed on a covered-to agent's station display instead of the originally-called EAS agent's LoginID extension.
Call Forwarding	Display VDN for Route-to DAC has no impact on the Call Forwarding feature.
Station Conference/ Transfer	When an EAS agent transfers or conferences a vector-initiated direct agent call that has the Display VDN for Route-to DAC feature activated to another agent or station user, the station display of the answering agent or station does not show the active VDN name that was previously displayed for the call. This is consistent with the existing station display treatment for transferred or conferenced calls that have a VDN name shown as the "to" party for a call.
VDN Override	Active VDN name station display rules for the VDN Override feature are applied to the Display VDN for Route-to DAC feature. For example, if an incoming trunk call is routed through a VDN where the VDN Override feature is enabled, and the call is routed to a second VDN by a route-to number vector step where the **Display VDN for Route-To DAC?** option is set to y, the station display for an EAS agent that receives a subsequent vector-initiated direct agent call shows the second VDN's name for the call instead of the called EAS agent's LoginID extension.

Table continues…

Interaction	Description
Redirect on No Answer (RONA)	The Display VDN for Route-to DAC feature is activated only for vector-initiated direct agent call to an EAS LoginID extension. When the RONA timer expires after the call is not answered, one of the following results occurs: • If subsequent vector processing again routes the call to an EAS LoginID extension by means of the Direct Agent Calling (DAC) feature, and the Display VDN for Route-to DAC feature is enabled, the active VDN name is shown on the covered-to agent station display. • If subsequent vector processing again routes the call to an EAS LoginID extension by means of the DAC feature, and the Display VDN for Route-to DAC feature is not enabled, then the EAS LoginID for the covered-to agent is shown on their station display.
Messaging systems for EAS Agents	The Display VDN for Route-To DAC feature has no interaction with messaging systems for a vector-initiated direct agent call that is routed to an EAS agent and subsequently covers to the agent's messaging-system mailbox.
Adjunct Routing	If a call is routed through a VDN having the Display VDN for Route-to DAC feature set to `y`, and an adjunct route vector step is executed that results in a direct agent call to an EAS agent, the active VDN name is displayed on the routed-to agent's station display instead of the called EAS agent's LoginID.

Call transfer to VDNs

Care needs to be taken when writing a vector to which callers will be transferred. This is especially true if the vector manipulates or tests data that is delivered with the incoming call, such as ANI, II-digits, or CINFO digits.

To understand why care is needed, it is necessary to understand how a transferred call is treated. There are three main steps in a call transfer.

1. The transferring party hits the transfer button. The caller is put on hold. A second call is created with the transferring party as the originator.
2. The transferring party dials the VDN extension. Vector processing starts. The transferring party, not the caller, hears the initial vector provided feedback, if any.
3. The transferring party hits the transfer button for the second time. The two calls merge. The transferring party is dropped from the call. The caller becomes the originator of the new call. The caller now begins to receive vector provided feedback.

Between transfer steps 2 and 3 there is always a small but finite amount of time during which it is the transferring party who is connected to the vector. Any testing of ANI, II-digits, or CINFO digits during this time window applies to the transferring party and not to the caller. For this reason, it is recommended that vectors not start with an ANI, II-digit, or collect cdpd/ced step. Insert a delay of sufficient length to allow the transferring party to complete the transfer.

A delay is not required before a collect x digits after announcement step because a collect announcement is restarted for the caller when the transfer is complete.

Call Vectoring interactions

Interaction	Description
AP Demand Print	A VDN cannot be used as an argument to the FAC for AP Demand Print.
Attendant Control of Trunk Group Access	If a **`route-to`** step in a vector dials a controlled trunk group, vector processing continues at the next step.
Attendant Recall	Attendant Recall to a VDN is blocked.
AUDIX Interface	A **`route-to`** step in a vector call the AUDIX extension. If a voice port can be seized to that adjunct, vector processing is terminated. The system sends a message to AUDIX requesting retrieval of messages for the originating extension (not the VDN). AUDIX also be accessed by the **`queue-to split`** and **`check split`** commands. Also, the messaging step use an AUDIX hunt group in its operation.
Authorization Codes	If authorization codes are enabled, and if a **`route-to`** command in a prompting vector accesses Automatic Alternate Routing or Automatic Route Selection and the VDN's Facility Restriction Level (FRL) does not have the permission to utilize the chosen routing preference, then no authorization code is prompted for and the **`route-to`** command fails.
Automatic Alternate Routing (AAR)/Automatic Route Selection (ARS)	Any **`route-to`** command in a vector can dial an AAR/ARS FAC followed by other digits. It cannot dial only the Facility Access Code.
Automatic Callback	Automatic Callback cannot be used for calls placed to a VDN.
Bridged Call Appearance	VDN extensions cannot be assigned to bridged appearance buttons. A **`route-to`** command to an extension with bridged appearances updates bridged appearance button lamps.
Busy Verification - Terminals, Trunks	Busy verification of VDNs is denied and intercept tone is returned.
Call Coverage	A VDN be administered as the last point in a coverage path.
Call Forwarding	Calls can be forwarded to a VDN. Calls placed by a **`route-to`** command to an extension that has call forwarding activated are forwarded. An attendant or phone with console permission cannot activate/deactivate call forwarding for a VDN. An attendant or phone with console permission cannot activate/deactivate call forwarding for a vector-controlled hunt group.

Table continues…

Interaction	Description
Call Detail Recording (CDR)	You can administer the Feature-Related System Parameters screen so that the VDN extension is used in place of the Hunt Group or Agent extension. This overrides the *Call to Hunt Group - Record* option of Call Detail Recording (CDR) for Call Vectoring calls. If a vector interacts with an extension or group that has Call Forwarding All Calls active, normal Call Forwarding/CDR interactions apply. For incoming calls to a VDN, the duration of the call is recorded from the time answer supervision is returned. If answer supervision is returned by the vector, and the call never goes to another extension, then the VDN extension is recorded as the called number in the CDR record. If the call terminates to a hunt group, then the VDN, hunt group, or agent extension is recorded as the called number as per the administration described above. If the call terminates to a trunk, then the following two CDR records are generated: • An incoming record with the VDN as the called number and the duration from the time answer supervision was provided to the incoming trunk. • An outgoing record containing the incoming trunk information as the calling number and the dialed digits and the outgoing trunk information as the called number. Outgoing vector calls generate ordinary outgoing CDR records with the originating extension as the calling number. No Ineffective Call Attempt records are generated for Call Vectoring **`route-to`** commands that are unsuccessful.
Call Detail Recording - Account Code Dialing	If a **`route-to number`** command in a vector specifies an CDR account code, vector processing continues at the next step.
Call Park	Calls cannot be parked on a VDN.
Call Waiting Termination	If an extension is busy and has call waiting termination administered, the **`route-to with cov n`** operation is treated as unsuccessful and vector processing continues at the next step. **`Route-to with cov y`** is successful (call will wait) and vector processing terminates.
Class of Restriction (COR)	Each VDN in the system has a Class of Restriction (COR) associated with it. This VDN COR is used to determine the calling permissions/restrictions, the AAR/ARS PGN, and the priority queuing associated with a vector.
Code Calling Access	A VDN cannot be used as the argument to the code calling access feature access code. If a **`route-to number`** command in a vector specifies the code calling feature access code, vector processing continues at the next step.
Conference	A call to a VDN can be included as a party in a conference call only after vector processing terminates for that call.

Table continues…

Interaction	Description
Data Restriction	Music will play on calls from data restricted extensions when the call receives music as the result of a wait-time vector step.
Facilities Restriction Level	If a **`route-to`** command dials an external number using AAR/ARS, the FRL associated with the VDN COR is used to determine the accessibility of a routing preference in an AAR/ARS pattern.
Facility Busy Indication	The facility busy lamp indication for a VDN is always off. A facility busy button can be used to call a VDN.
Facility Test Calls	If a **`route-to number`** command in a vector specifies a Facility Test Call, vector processing continues at the next step.
Forced Entry of Account Codes	If a COR requiring entry of account codes is assigned to a VDN, the **`route-to number`** commands executed by the associated vector are unsuccessful and vector processing continues at the next step.
Individual Attendant Access	A call sent to an attendant by a **`route-to number`** command can wait in the attendant priority queue. The call is removed from vector processing.
Integrated Directory	VDN names and extensions are not available in the Integrated Directory feature.
Intercept Treatment	A VDN cannot be used for Intercept Treatment.
Inter-PBX Attendant Calls	A **`route-to number`** command in a vector can dial the Inter-PBX Attendant. If the call attempts to access a controlled trunk group, vector processing continues at the next step.
Intraflow and Interflow	The functionality of intraflow and interflow can be obtained using the **`check`** and **`goto`** Call Vectoring commands. Calls can intraflow from an ACD split or skill that is not vector-controlled into one that is vector-controlled.
Leave Word Calling (LWC)	Leave Word Calling (LWC) messages cannot be stored, canceled, or retrieved for a VDN.
Night Service	A VDN can be administered as a night service destination. **`Route-to`** commands that route to destinations with night service activated redirect to the night service destinations.
Priority Calling	A VDN cannot be used with the priority calling access code. Intercept tone is supplied to the user. If a **`route-to number`** in a vector specifies the priority calling access code, vector processing continues at the next step.
Property Management System Interface	VDNs cannot be used with the following features and functions: Message Waiting Notification, Check-In, Check-Out, Room Status, and Automatic Wakeup.
Recorded Announcement	The first announcement extension, second announcement extension, first announcement delay, second announcement delay, and recurring second announcement do not exist for a vector-controlled hunt group.
Redirection on No Answer	If an ACD split or skill or direct agent call is not answered after an administered number of rings, RONA can redirect that call to a VDN for alternate treatment.

Table continues…

<table>
<tr><th>Interaction</th><th>Description</th></tr>
<tr><td>Ringback Queuing</td><td>External call attempts made using <code>route-to</code> commands with coverage no are not queued using Ringback Queuing when all trunks are busy. External call attempts made using <code>route-to</code> commands with coverage yes are.</td></tr>
<tr><td>Route Calls to Agent by skill level</td><td> Using the skill level preference parameters, you can request a preference to route calls to an available (idle) agent who has a specified skill level or is in a skill level range.
This option is specified by using the skill level preference parameters on the <code>check</code> Vectoring command. In an agent surplus condition (available-agents > 0), you can request that the call is routed to an available agent who has a specific skill level or a skill level within a specified range.
If you select pref-level, the system displays only the Skill Level1 field, where you can enter a skill value of 1-16. If you select pref-range, the system displays two fields, Skill Level1 and Skill Level2. Using these two fields, you can enter a range of preference levels, such as 5 (Skill Level1) to 13 (Skill Level2). The values in both these fields need to be between 1 and 16. The number in Skill Level2 field needs to be equal to or greater than the number you enter in the Skill Level1 field.

<code>Skill</code>: Skill level preference options are only available for the check skill version of the command, not for check split or check best.
<code>if available-agents > 0 or greater</code>: This option ensures that the skill level preference is applied only in agent surplus conditions.
<code>all-levels</code>: The system ignores the skill level of the agent. This is the default value.
<code>pref-level</code>: The system displays the Skill Level1 field in which you can enter a skill level value for the agent from 1 to 16. Preference level of 1 is the best while 16 is the least.
<code>pref-range</code>: The system displays two fields, Skill Level1 and Skill Level2. Using these two fields, you can enter a range of preference levels, such as 5 (Skill Level1) to 13 (Skill Level2). The values in both these fields need to be between 1 and 16. The number in Skill Level2 field needs to be equal to or greater than the number you enter in the Skill Level1 field.

Following is a sample vector command:
<pre>check skill 5 pri h if available-agents > 0 pref-range 1 to 3
queue-to skill 17 pri t</pre>
For example, in agent surplus conditions, this feature allows you to route high-value and critical calls to the best skilled agents for this skill and route the low valued calls to trainees or novice agents who have the assigned same skill.
For more details on syntax and usage of check skill command and skill level preference parameters, see the Programming Call Vectoring Features in Avaya Aura® Call Center Elite document. </td></tr>
</table>

Table continues…

Interaction	Description
Send All Calls	If the destination of a `route-to with coverage no` command has the Send All Calls (SAC) feature active, calls are not redirected. If there is an idle appearance, the call terminates and vector processing stops. If not, vector processing continues at the next step. If the **send-calls** button is pressed after a vector call is terminated, button activation is denied.
Time of Day Routing	Since a `route-to number` command in a vector can specify the Automatic Alternate Routing (AAR) or Automatic Route Selection (ARS) access codes, the Time-of-Day (TOD) routing algorithm can be used to route the call.
Timed After Call Work (ACW) Associated field: **After Xfer or Held Call Drops**	A Timed After Call Work (TACW) interval can be assigned to a VDN. For incoming ACD or DAC calls, an Auto-In agent is placed into Timed ACW mode, instead of immediately making the agent available, if the caller drops a held call or the agent transfers the call. You can enable this feature for the agents in a hunt group or for calls delivered from a VDN when the Timed ACW Interval field is set to a non-0 value.
Timed Reminder	The attendant Timed Reminder is not available for calls placed, transferred, or extended to a VDN. Vectoring causes all other timers to be ignored.
Transfer	Calls can be transferred to a VDN.
Traveling Class Mark	A TCM is sent when a `route-to` command dials a seven-digit Electronic Tandem Network (ETN) or 10-digit Direct Distanced Dialed number using AAR/ARS. This TCM is the FRL associated with the VDN Class of Restriction (COR).
VDN in a Coverage Path (VICP)	A call covering to a VDN can be routed to any valid destination by the `route-to` command. You can enable the **Cvg Enabled for VDN Route-to Party** field on the Coverage Path screen to allow a second redirection when a `route-to` command with cov = y vector step processes a previously covered call routed to a VDN. The default is `n` which retains the single redirection operation. When the `route-to` command terminates a covered call locally, information about the principal and the reason for redirection is retained with the call. This information is displayed on the phone, or passed to an AUDIX or a Message Center system. The COR assigned to a VDN determines the PGN. The PGN in turn determines the AAR or ARS routing tables used by `route-to` commands. When a call covers to a VDN, VDN Override rules have no effect on the display shown on an answering display telephone. This station shows the normal display for a covered call.

Call Vectoring and BCMS/CMS interactions

Call Vectoring interacts with a management information system that helps monitor and report Call Vectoring activities. The management system is either an Avaya Call Management System (CMS) or an Avaya Basic Call Management System (BCMS).

CMS, which is installed on an adjunct processor, collects and processes ACD information to generate reports. BCMS, which is part of Communication Manager, collects ACD information and generates a limited number of reports. CMS reporting and data storage capabilities are much more extensive than those of BCMS.

Chapter 4: Call Vectoring examples

Example	Feature
Customer service center on page 256	Basic Call Vectoring
Automated attendant on page 257	Call Prompting
Data in or voice answer and data or message collection on page 258	Call Prompting and Basic Call Vectoring
Distributed call centers on page 260	Look-Ahead Interflow (LAI) and Basic Call Vectoring
Help desk on page 262	Adjunct Routing, Call Prompting, and Basic Call Vectoring
Insurance or service agency on page 262	Basic Call Vectoring, Call Prompting, Rolling Average Speed of Answer (ASA), Expected Wait Time (EWT), Vector Directory Number (VDN) calls, and Automatic Number Identification (ANI) routing
Warranty service with EAS on page 265	Basic Call Vectoring and Expert Agent Selection (EAS)
Resort reservation service on page 270	Basic Call Vectoring, Adjunct Routing, Call Prompting, and EAS
Local attendant group access code on page 273	Attendant Vectoring
Incoming trunk calls to attendant group on page 274	Attendant Vectoring
Incoming LDN calls on page 274	Attendant Vectoring
QSIG Centralized Attendant Service on page 275	Attendant Vectoring
Night station service on page 276	Attendant Vectoring
Holiday Vectoring on page 277	Holiday Vectoring
Network Call Redirection on page 279	Multisite Best Service Routing (BSR) and NCR
Best Service Routing using Expected Wait Time and agent adjustments on page 281	Multisite BSR
Dial by Name on page 283	Basic Call Vectoring and Call Prompting

Customer service center

In the example, a customer service center is open on weekdays from 8 a.m. to 5 p.m. The center has two separate phone numbers for regular and priority customers. The following vector examples show how calls to the customer service center are handled.

Example application - customer service center

```
VDN (extension=1021  name=Customer Serv  vector=21)
Vector 21:
      1. goto vector 29 @step 1 if time-of-day is all 17:00 to all 08:00
      2. goto vector 29 @step 1 if time-of-day is fri 17:00 to mon 08:00
      3. goto step 10 if calls-queued in split 1 pri l > 10
      4. queue-to split 1 pri m
      5. wait-time 10 seconds hearing ringback
      6. announcement 3521
      7. wait-time 50 seconds hearing music
      8. announcement 3522
      9. goto step 7 if unconditionally
     10. busy

VDN (extension=1022  name=Priority Cust  vector=22)
Vector 22:
      1. goto vector 29 @step 1 if time-of-day is all 17:00 to all 08:00
      2. goto vector 29 @step 1 if time-of-day is fri 17:00 to mon 08:00
      3. goto step 12 if calls-queued in split 1 pri h > 10
      4. queue-to split 1 pri h
      5. announcement 3521
      6. wait-time 10 seconds hearing music
      7. check split 2 pri h if oldest-call-wait < 20
      8. check split 3 pri h if oldest-call-wait < 20
      9. announcement 3522
     10. wait-time 60 seconds hearing music
     11. goto step 7 if unconditionally
     12. route-to number 0 with cov n if unconditionally

No VDN
Vector 29:
      1. announcement extension 3529
      2. wait-time 10 seconds hearing silence
      3. disconnect after announcement 3529
```

When a *priority customer* places a call, the system accesses Vector 22. The first two steps of Vector 22 determine if the call arrives during non business hours. If the call arrives between 5:00 p.m. and 8:00 a.m. on any given day, step 1 routes the call to Vector 29. Step 2 does the same if the call arrives during the weekend, that is, between 5:00 p.m. Friday and 8:00 a.m. Monday. If the system accesses Vector 29, the caller hears an announcement twice and the call is then disconnected.

If the call is placed during business hours, step 3 of Vector 22 determines if the number of high-priority calls that are queued in the main split exceeds 10. If more than 10 calls are in queue, control is sent to step 12, which routes the call to an attendant. If less than 10 calls are in queue, the call is queued to the main split as indicated in step 4. An appropriate announcement plays (step 5), followed by a wait period in step 6.

If the call is not answered after the wait time specified in step 6, steps 7 and 8 attempt to queue the call to a backup split (splits 2 and 3, respectively). The call is queued to either split if the oldest call in the split has been waiting fewer than 20 seconds.

Even if the call is queued to one of the backup splits, the call is passed to steps 9 through 11, which implement an announcement-wait cycle that continues until either an agent answers the call, or the caller abandons the call.

When a *non priority customer* places a call, *vector 21* is accessed. Vector 21 provides a treatment similar to that provided by Vector 22, with the following exceptions:

- Backup splits are not queried for non priority calls
- Priority calls are assigned a higher priority in the queue
- Priority calls route to an operator (step 12 of Vector 22) when too many calls are queued, but non priority calls route to a busy signal (step 10 in Vector 21).

Automated attendant

With Automated Attendant, a caller can enter the extension of the agent. The caller can enter up to 16 digits from a touch-tone telephone.

Call centers use Automated Attendant if call centers do not have DID trunks and if the callers know the extension of the agents. Automated Attendant reduces call center costs, as call centers do not require live attendants.

Example application - automated attendant

```
 1. wait-time 0 seconds hearing ringback
 2. collect 5 digits after announcement 30001 [You have reached Ridel Publications.
Please dial a 5-digit extension or
    wait for an attendant.]
 3. route-to digits with coverage y
 4. route-to number 0 with cov n if unconditionally
 5. stop
```

Step 1 of the vector contains the **wait-time** command, which is placed before the **collect digits** command in step 2 to provide the caller with ringback in the event that a TouchTone Receiver (TTR) is not immediately available. A TTR must be connected in order for the **collect digits** command to take effect. Once a TTR is connected, the caller is prompted to enter the destination extension of the party, that is, step 2. The **collect digits** command in step 2 collects the digits. Thereafter, the **route-to digits** command in step 3 attempts to route the call to the destination.

If the **route-to digits** command fails because the caller fails to enter any digits, or enters an invalid extension number, the **route-to number** command in step 4 routes the call to an attendant. However, as long as the destination is a valid extension, the **route-to digits** command succeeds, coverage applies, and vector processing terminates.

Note:

Even if the destination is busy, vector processing terminates because coverage call processing takes effect.

Data in or voice answer and data or message collection

The scenario involves a mutual fund company that is open 24 hours a day, 7 days a week. All incoming calls are directed to a single VDN extension that maps to a main vector. The main vector presents a menu of options to the calling party and the vector uses Call Prompting to determine the desired service. The following three services are offered to the customers:

- New Account: To open new accounts.
- Account Inquiry: To make inquiries about respective accounts.
- Net Asset Value (NAV): To receive information on the NAV of the company funds.

If the caller selects *account inquiries*, the caller is prompted to enter the account number. The agent can use the **callr-info** button to view the account number.

Note:

If the agent has a two-line display telephone, the account number is automatically displayed on the second line. Some supported display telephones include 6416, 6424, 8410, 8434 and CallMaster® set.

The scenario includes the use of the following three applications supported by Call Prompting:

- Data In/Voice Answer (DIVA): The caller can receive information on a topic selected at the prompt. The caller selects the desired topic by entering the appropriate digits.
- Data Collection: Collects the digits entered by a caller. The digits are all numeric and one such example is an account number or US Social Security Number.
- Message Collection: The caller can leave a recorded message instead of waiting for the call to be answered.

The following four vectors illustrate how the mutual fund company handles telephone calls. The vector is programmed to check if queue slots are available.

Example application - mutual fund company

```
VDN (extension=1030   name=ABC Inv   vector=10   display override=y)
Vector 10
      1. wait-time 0 secs hearing ringback
      2. collect 1 digits after announcement 3531 [Thank you for calling ABC
Investments. If you wish to open a new account,
         please dial 1. If you wish to make an account inquiry, please dial 2. If you
wish to know the current net asset
         values of our funds, please dial 3.]
      3. route-to number 1031 with cov y if digit = 1
      4. route-to number 1032 with cov y if digit = 2
      5. route-to number 1033 with cov y if digit = 3
      6. route-to number 0 with cov n if unconditionally
      7. disconnect after announcement none

VDN (extension=1031   name=New Account   vector=11)
Vector 11
      1. goto step 5 if calls-queued in split 1 > 19
      2. queue-to split 1 pri t
      3. announcement 3535
      4. wait-time 10 secs hearing music
      5. collect 1 digits after announcement 4020 [We are sorry. All our operators are
busy at the moment. Dial 1 to leave
```

```
         your name and telephone number.]
      6. goto step 10 if digit = 1
      7. announcement 3537
      8. wait time 50 secs hearing music
      9. goto step 6 if unconditionally
     10. messaging split 5 for extension 4000
     11. announcement 3538 [We are sorry, we cannot take your message now. Please hold
or call back later.]
     12. goto step 4 if unconditionally
```

DIVA and data/message collection vector example

```
VDN (extension=1032    name=Account Inq   vector=12)
Vector 12:
      1. wait-time 0 secs hearing ringback
      2. collect 6 digits after announcement 3533 [Please enter your 6-digit account
number.]
      3. goto step 7 if calls-queued in split 1 > 19
      4. queue-to split 1 pri m
      5. announcement 3535
      6. wait-time 60 secs hearing music
      7. collect 1 digits after announcement 4020 [We are sorry. All our operators are
busy at the moment. Dial 1 to leave
         your name and telephone number.]
      8. goto step 12 if digit = 1
      9. announcement 3537
     10. wait time 50 secs hearing music
     11. goto step 8 if unconditionally
     12. messaging split 5 for extension 4000
     13. announcement 3538 [We are sorry, we cannot take your message now. Please hold
or call back later.]
     14. goto step 4 if unconditionally

 VDN (extension=1033   name=Net Asset Val   vector=13)
 Vector 13:
      1. disconnect after announcement 3534 [The net asset values of our funds
at the close of the market on Wednesday, May 15         were as follows: ABC
Growth.....33.21.....up 33 cents; ABC High Yield.....11.48.....down 3 cents.]
```

When the call is placed, vector processing begins in vector 10, which is the main vector. Step 1 of the vector contains the **`wait-time`** command, which is placed before the **`collect digits`** command in step 2 to provide the caller with feedback in the event that a tone detector is not immediately available. Once a tone detector is connected, the **`collect digits`** command provides an announcement that requests the caller to enter 1, 2, or 3, depending on the service. If the caller enters a digit other than 1, 2, or 3, or if the caller fails to enter any digits within 10 seconds, the command fails and the call is routed to an attendant (step 6). If the caller enters 1, 2, or 3 within 10 seconds, the call is routed to the vector specified in the **`route-to number`** command, which appears in steps 3, 4, and 5.

For instance, when prompted, the caller enters 3 to learn about the NAV of the company funds. The **`route-to number`** command in step 3 and in step 4 fail, because in each case, the digit that is tested for in the condition portion of the command is not 3. However, the **`route-to number`** command in step 5 succeeds because the digit that is tested matches the one entered by the caller. Accordingly, the call is routed to VDN extension 1033 and vector processing continues in vector 13.

The **`announcement`** command in step 1 of vector 13 provides the caller with the information on NAV and disconnects the call.

The process just described, wherein the caller receives information after entering a digit at the prompt, is an example of the DIVA application.

Returning to the main vector, if caller wants to make an inquiry and enters 2 when prompted, step 3 fails, but step 4 succeeds. Accordingly, the call is routed to VDN extension 1032 and vector processing continues in vector 12.

The **`collect digits`** command in step 2 of vector 12 first requests the caller to enter the 6-digit account number. The command then collects the digits. Whether or not the caller enters the digits correctly, the **`queue-to split`** command in step 4 queues the call. If an agent does not immediately answer the call, the standard announcement in step 5 plays and a delay is provided in step 6. The announcement in step 7 provides the caller with the option to either leave a message or hold. The caller is prompted to enter 1 to leave a recorded message. If the caller does not enter 1, the **`goto step`** command in step 8 fails and an announcement-wait cycle is implemented by steps 9, 10, and 11 until the call is answered or abandoned. If the caller does enter 1 within 10 seconds, step 8 passes control to step 12. The **`messaging split`** command in step 12 attempts to connect the caller to an AUDIX or a message center split. If the connection is made, the caller hears ringback and can then leave a message. If the connection is not made, the step is unsuccessful and step 13 provides an announcement that indicates that a connection is not made. Thereafter, the **`goto step`** command in step 14 sends the call control back to step 6, which leads the caller back to the steps to leave a message.

The process just described, wherein the caller enters digits that comprise an official number, is an example of the Data Collection application. If the agent has a **callr-info** button or a two-line display, the agent can see the digits entered by the caller. As a result, the agent does not have to request the caller for an account number.

Finally, if another caller wants to open an account and enters 1 when prompted in the main vector, step 3 of the main vector is successful. Accordingly, the call is routed to VDN extension 1031 and vector processing continues in vector 11.

In step 2 of vector 11, the call is queued to the main split. Thereafter, step 3 provides an appropriate announcement and step 4 provides a delay period. The announcement in step 5 provides the caller with the option to either leave a recorded message or hold. This is an example of the Message Collection application. The caller is prompted to enter 1 to leave a recorded message. If the caller does not enter 1, the **`goto step`** command in step 6 fails and an announcement-wait cycle is implemented by steps 7, 8, and 9 until the call is answered or abandoned. If the caller does enter 1 within 10 seconds, step 6 passes control to step 10. The **`messaging split`** command in step 10 attempts to connect the caller to an AUDIX or message center split. If the connection is made, the caller hears ringback and can then leave a message. If the connection is not made, the step is unsuccessful and step 11 provides an announcement that indicates that a connection is not made. Thereafter, the **`goto step`** command in step 12 sends call control back to step 4, which leads the caller back into the steps to leave a message.

Distributed call centers

The example involves two call centers located in New York and Denver. Calls to the New York call center are queued to up to two splits. If calls remain unanswered for a period of time, a Look-Ahead Interflow (LAI) call attempt is made to the Denver call center. If there are less than 10 queued calls in Denver, the LAI call attempt is accepted and serviced at Denver. Else, the call is denied and remains in queue in New York until an agent becomes available. The two vectors below illustrate the process.

Example application - distributed call centers

```
Sending Communication Manager:
VDN (extension=1080   name=New York Office   vector=80)
Vector 80:
       1. goto step 11 if calls-queued in split 1 pri m > 5
       2. queue-to split 1 pri m
       3. announcement 3580 [All our agents are busy at the moment. Please hold.]
       4. wait-time 6 seconds hearing music
       5. route-to number 913035661081 with cov n if unconditionally
       6. check split 2 pri m if calls-queued < 5
       7. wait-time 6 seconds hearing music
       8. announcement 3581 [All our agents are still busy. Please hold.]
       9. wait-time 60 seconds hearing music
      10. goto step 5 if unconditionally
      11. busy

Receiving Communication Manager:
VDN (extension=1081   name=Denver Inflow   vector=81)
Vector 81:
       1. goto step 7 if calls-queued in split 3 pri l > 10
       2. wait-time 0 seconds hearing music
       3. queue-to split 3 pri h
       4. announcement 3582 [We apologize for the delay. Please hold.]
       5. wait-time 60 seconds hearing music
       6. goto step 5 if unconditionally
       7. disconnect after announcement none
```

Vector 80 is on the sending Communication Manager from a call center in New York, while vector 81 is on the receiving Communication Manager at a call center in Denver.

In the sending Communication Manager, the call is queued to split 1 at a medium priority (step 2) if the condition in step 1 is met. If the condition is not met, the call is routed to busy in step 11.

If the call is queued, but not immediately answered, the caller hears an announcement (step 3) and music (step 4). If the call is still not answered at this point, step 5 places a LAI call attempt to the receiving Communication Manager, on which vector 81 resides.

Step 1 in the receiving Communication Manager determines whether the call can be serviced in Denver. If the number of calls queued at any priority in split 3 is greater than 10, vector 81 does not service the call. In such a case, control is passed to step 7, which rejects the LAI call attempt. However, if the test in step 1 succeeds, the call is queued by the receiving Communication Manager in split 3 at a high priority (step 3) and the LAI call attempt is accepted. Accordingly, the call is removed from the main split queue in New York and the control is passed to the Denver Communication Manager, where vector processing continues at step 4.

If the receiving Communication Manager does not accept the LAI call attempt, control is passed to step 6 of the sending Communication Manager. The step then queues the call to split 2 at a medium priority, provided there are less than five calls queued in that split. Thereafter, the customary announcement-wait sequence plays (steps 7, 8, and 9). Finally, if necessary, step 10 sends the control back to step 5, which makes another LAI attempt and the cycle is repeated.

Note:

To prevent confusion, the treatment provided at the receiving Communication Manager must be consistent with the treatment provided at the sending Communication Manager. In the distributed call centers example, note that the caller hears music and never ringback or silence, at the sending Communication Manager. Accordingly, music must be featured at the receiving Communication Manager.

Help desk

The scenario involves a help desk at a computer firm. The help desk is configured into three groups. One group handles hardware problems, the second group handles software problems, and the third group handles general problems. The information provided in the Adjunct Switch Application Interface (ASAI) route request, that is, calling party number, called number, and collected digits, is used to route the call to an appropriate agent. Such an agent can be the one who last serviced the caller, or the next available agent for the specific caller. Also, based on the traffic conditions of Communication Manager and the caller entered digit, the call can be diverted to other destinations such as other ACD splits, announcements, or to another Communication Manager.

Example application - help desk

```
1. wait-time 0 seconds hearing ringback
2. collect 1 digits after announcement 4704 [Welcome to the TidyBits Computer
Corporation help desk. For hardware problems,
   dial 1. For software problems, dial 2. For general queries, dial 3.]
3. adjunct routing link 12
4. wait-time 4 seconds hearing ringback
5. route-to number 3710 with cov y if digit = 1
6. route-to number 3720 with cov y if digit = 2
7. route-to number 3730 with cov y if digit = 3
8. route-to number 0 with cov n if unconditionally
9. stop
```

Step 1 of the vector contains the **wait-time** command to provide the caller with ringback in the event that a TTR is not immediately available. A TTR must be connected for the **collect digits** command to take effect. In step 2 of the vector, the caller is prompted to enter 1, 2, or 3, based on the type service required by the caller. Thereafter, the **adjunct routing link** command in step 3 instructs Communication Manager to send a route request to the adjunct processor. The route request contains the called party number, the calling party number, and the digit that is collected in step 2, along with the other pertinent information for adjunct routing. If 1, 2, or 3 is not entered and if the adjunct does not return a route, the call is eventually routed to an attendant (step 8).

If the **adjunct routing link** command in step 3 succeeds, the adjunct uses the information included in the route request to select the appropriate route for the call. If the caller enters 1 and the **adjunct routing link** command succeeds, the call is routed to agent who handles hardware related queries. For general hardware problems, the call can be routed to a larger group of ACD agents before the call is queued.

Insurance or service agency

In the example, an insurance company call center handles calls from independent field agents, policy holders with claims, policy holders with need for customer service, and several general service agency type toll–free number client accounts such as 800, 888, 877, 866 number types. Each different type of call has an 800 number that routes the calls to associated VDNs.

Call center requirements:

- Independent field agents require prompt service. The field agents call the company to find the latest rates for specific clients, to set up policies, or to make adjustments. Therefore the insurance company wants to maintain an Average Speed of Answer (Rolling-ASA) of less than 30 seconds for field agent calls. Field agent calls are given the highest priority in queue.
- Calls to Claims must be separated by area code. Training for the Claims agents is based on the area of the country for the claim. A particular group of agents can be given training for more than one area code. Therefore, area codes must not be tested individually and can be grouped in Vector Routing Tables (VRT).
- The insurance company wants the customer service callers to hear an announcement indicating the wait duration for a service.
- The insurance company also sells spare call center capacity to client accounts. The account contracts are provided on the basis that only so many calls to a particular account are accepted at any given time.

In this example, Rolling ASA routing is used to maintain the rolling ASA objective of less than 30 seconds for field agent calls. ANI or CLID routing is used to partition arriving calls based on any of the following combinations so as to route calls to the appropriate Claims agent:

- Area code, that is, Numbering Plan Area (NPA)
- Area code and Central Office (CO) within the NPA, that is, NPA-NXX
- Entire phone number, that is, NPA-NXX-XXXX

The terms ANI and CLID are, in practice, used interchangeably with respect to vectors. ANI equates to the caller's number when the caller dials a toll-free number, whereas CLID equates to the caller's number when the caller dials a non toll-free number. EWT routing is used to notify callers about the expected wait time if the wait time is more than 60 seconds. VDN Calls routing is used to regulate the number of calls to service agency clients.

The following table shows the VDNs and vectors associated with each type of call.

VDN table for insurance agency or service agency example		
Type of service	**VDN number**	**Vector number**
Field Agents	1001	1
Claims	1002	2
Customer Service	1003	3
Client 1	1004	4
Client 2	1005	5

Note:

To demonstrate the features described in the example, the sample vectors do not include tests for unstaffed or full queues and out-of-hours operation.

Step 1 queues the call to the main split. If the main split is currently answering calls within the target time of 30 seconds, step 2 bypasses all the backup splits and goes to the announcement in step 6. The call is handled by split 10 within the time constraints. However, if the call is not answered by the time that vector processing reaches step 8, the backup splits are checked.

If the Rolling ASA for the main split is more than 30 seconds, steps 3, 4, and 5 check backup splits are executed. The call is queued to any of the splits that have a Rolling ASA of less than 30 seconds. If the call still is not answered by the time vector processing reaches step 8, the backup splits are checked again.

Use the following vector example to route Claims calls by area code.

Claims vector example

```
 VDN 1002 -- Claims calls
 1. goto step 10 if ani = none
 2. goto vector 21 @step 1 if ani = 201+
 3. goto vector 22 @step 1 if ani = 212+
 4. goto vector 23 @step 1 if ani in table 1
 5. goto vector 24 @step 1 if ani in table 2
 6. goto vector 25 @step 1 if ani in table 3
 7. goto vector 26 @step 1 if ani in table 4
 8. goto vector 27 @step 1 if ani in table 5
 9. goto vector 30 @step 1 if unconditionally
10. wait-time 0 seconds hearing ringback
11. collect 3 digits after announcement 10001   [Please dial your area code]
12. goto vector 30 @step 1 if digits = none
13. goto vector 21 @step 1 if digits = 201+
14. goto vector 22 @step 1 if digits = 212+
15. goto vector 23 @step 1 if digits in table 1
16. goto vector 24 @step 1 if digits in table 2
17. goto vector 25 @step 1 if digits in table 3
18. goto vector 26 @step 1 if digits in table 4
19. goto vector 27 @step 1 if digits in table 5
20. goto vector 30 @step 1 if unconditionally
```

Each VRT referenced in the example contains a list of area codes with the “+” wildcard. Each list of area codes is handled by a specific group of agents. Vectors 21 through 27 queue calls to the appropriate group of agents. Vector 30 provides a live agent to screen calls that have area codes that are not listed in any table or vector step. The vector also provides access to an agent when ANI is not available and the caller does not enter an area code when prompted.

The following vector example notifies customer service callers of the expected wait time.

Customer Service vector example

```
 VDN 1003 -- Customer Service calls
 1. goto step 10 if expected-wait for split 32 pri l > 600
 2. queue-to split 32 pri l
 3. wait-time 20 seconds hearing ringback
 4. goto step 8 if expected-wait for call > 40
 5. announcement 1100
 6. wait-time 40 seconds hearing music
 7. goto step 5 if unconditionally
 8. converse-on split 80 pri l passing wait and none
 9. goto step 5 if unconditionally
10. disconnect after announcement 1400
```

In step 1, callers who wait more than 10 minutes are routed to a call back later announcement. Step 4 routes callers to a VRU to be given the EWT announcement while the callers hold the place in the queue.

The following vector examples can be used to regulate the number of calls to service agency clients. In this example, client 1 has contracted for 100 simultaneous calls while client 2 has contracted for only 50 simultaneous calls.

Service Agency Clients vector examples

```
VDN 1004-- Client 1 calls
1. goto step 3 if counted-calls to vdn 1004 <= 100
2. busy
3. queue-to split 60 pri l
4. wait-time 20 seconds hearing ringback
5. announcement 12000
6. wait-time 60 seconds hearing music
7. goto step 5 unconditionally

VDN 1005 -- Client 2 calls
1. goto step 3 if counted-calls to vdn 1005 <= 50
2. busy
3. queue-to split 60 pri l
4. wait-time 20 seconds hearing ringback
5. announcement 12000
6. wait-time 60 seconds hearing music
7. goto step 5 unconditionally
```

In both the examples vectors, the first step routes calls to queue if the number of contracted calls is not exceeded. Otherwise callers receive a busy signal.

Warranty service with EAS

The scenario involves a major appliance company that offers one year warranties and extended warranties on major appliances such as dishwashers, refrigerators, washers, and dryers. The warranties are printed in English and Spanish to serve customers who speak both languages. 800 numbers are provided for calling both English-speaking agents and Spanish-speaking agents. Bilingual agents with Spanish-speaking skills are hired to back up the groups of English-speaking agents. Agents are first trained on all appliance models of a certain type and then on all appliance models for a room such as the kitchen, and the laundry room.

The skills shown in the following table are required for the warranty service call center.

Skill for a warranty service call center		
Appliance type	**English skill number**	**Spanish skill number**
Kitchen appliances	10	20
Dishwashers	11	21
Refrigerators	12	22
Laundry appliances	30	40
Washers	31	41
Dryers	32	42
Supervisors	100	

The VDN skill preferences are set up as shown in the following table.

VDN skill for the warranty service call center					
VDN skill preference	**Appliance**	**VDN**	**First**	**Second**	**Third**
English	Dishwasher	1100	11	10	20
	Refrigerator	1101	12	10	20
	Washer	1102	31	30	40
	Dryer	1103	32	30	40
Spanish	Dishwasher	1200	21	20	--
	Refrigerator	1201	22	20	--
	Washer	1203	41	40	--
	Dryer	1204	42	40	--

The agent skills are set up as shown in the following table.

Agent skills for the warranty service call center				
Agent	**Skill level 1**		**Skill level 2**	
Kim	42	40	41	30
Michelle	100	--	--	--
Beth	31	--	--	--
Mike	32	--	30	--

Once skills are assigned to VDNs and to agents, calls are directed to the appropriate vector.

The goal of the warranty service call center is to answer 80 percent of the incoming calls within 20 seconds. Accordingly, if a call directed to a vector is not answered by the time the announcement finishes, a second group of agents is viewed, enlarging the agent pool. If the call is not answered within the following 10 seconds, a third group of agents is viewed.

Since the call center has only a few bilingual agents, the management wants to reserve the agents for Spanish-speaking callers only. This can be done by giving Spanish-speaking callers a higher priority in the vector or by assigning a higher skill level to Spanish skills. Also, if a Spanish-speaking caller waits more than 30 seconds for service, a supervisor of the Spanish-speaking skills takes the calls.

The figures depict the setup for the warranty service call service. Specifically, the figures show the vectors and call flows for callers with a broken washer or dryer who need service. Separate vectors are used to provide an announcement in Spanish and in English, see step 2. The same two vectors can be used for callers who need service for broken dishwashers and refrigerators.

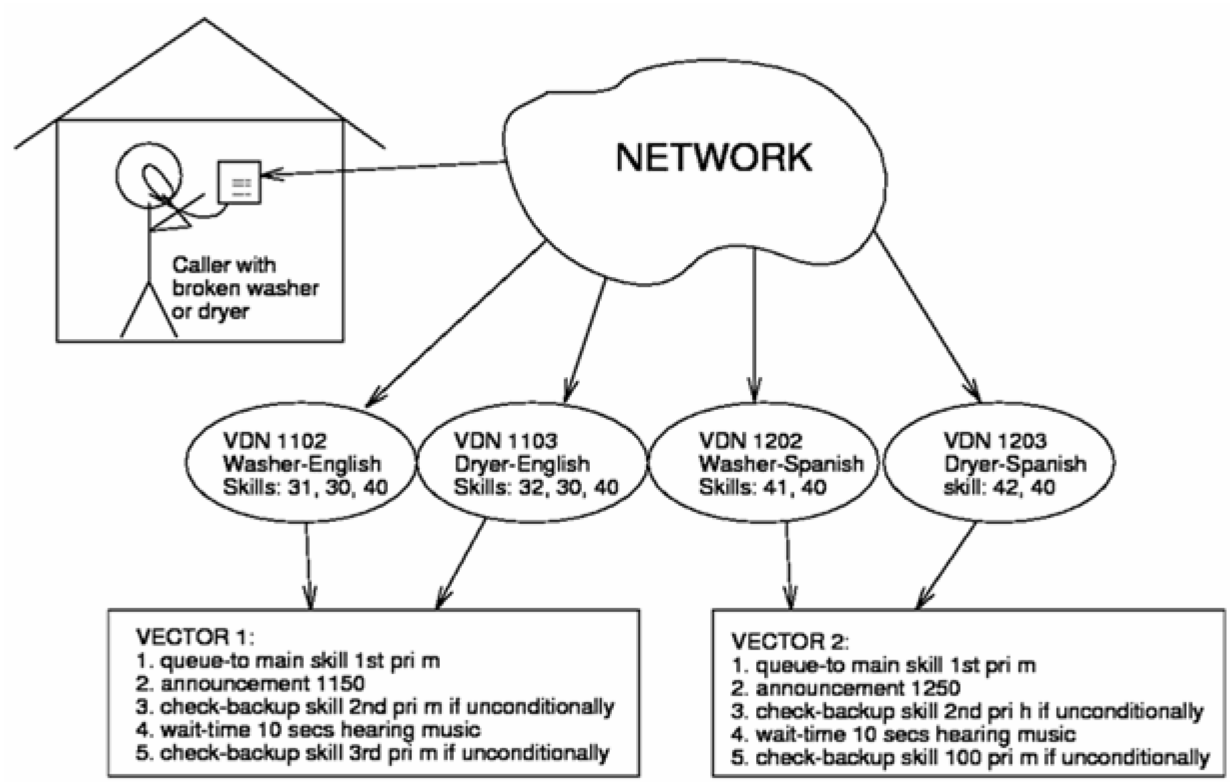


Figure 1: Warranty service call center (part 1)

The next figure depicts how the vector-processed call is directed to the appropriate call queue. The figure also shows how the call is directed to the appropriate agent or agents. The agent skills are indicated below the name of each agent. Dashed lines indicate backup or secondary skills.

Note:

Only a small sample of agents is shown in the example figure.

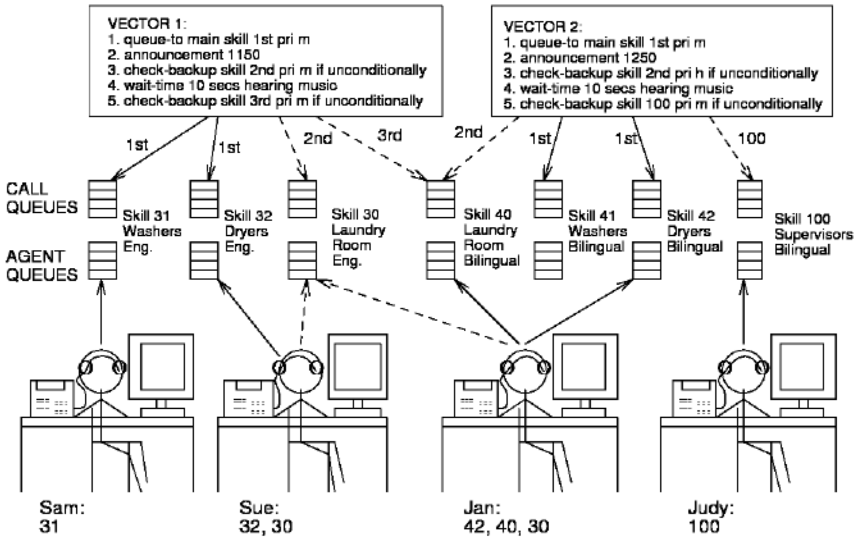


Figure 2: Warranty service call center (part 2)

A Spanish-speaking customer has a broken dryer. The caller dials the call center number. Communication Manager receives and directs the call to VDN 1203, which points to vector 2. The system administrator assigns the VDN skill preferences 42 for dryers and 40 for laundry appliances as the first and second skill preferences respectively.

Once vector processing starts, the **`queue-to skill`** command in step 1 of vector 2 queues the call to the skill group corresponding to the first VDN skill (42-Dryers Bilingual). If an agent with skill 42 is available, the agent answers the call. If the agent is not available, an appropriate delay announcement in step 2 plays. Next, the **`check skill`** command in step 3 attempts to queue the call to the skill group corresponding to the 2nd VDN skill (40-Laundry Appliances Bilingual). If an agent with skill 40 is available, the agent answers the call. Otherwise, a wait period is provided in step 4 and the **`check skill`** command in step 5 checks the specific skill (100-Supervisors Bilingual) for available agents.

Notify callers about queue position

XYZ call center has the following requirements:

- Announce the position of a call in queue to callers.
- Do not use a wait time estimate because the call traffic for this call center is random and the talk times are variable.
- Do not use the existing Integrated Voice Response (IVR) or Voice Response Unit (VRU) equipment.

Scenario solution

The XYZ call center decides to use the interflow-qpos goto step conditional to test the caller position in queue. The interflow-qpos goto conditional checks the caller's position in queue from 1 (next in line) to 9 (8 calls are ahead).

Scenario prerequisites

- Virtual Routing (LAI) must be active.
- Set the **Interflow-Qpos EWT Threshold** field on the Feature-Related System Parameter screen to `0` seconds. The interflow-qpos tests what is defined as the eligible queue. Setting the field to `0` does not exclude calls at the top of the queue.

How to set up the interflow-qpos conditional

- Use the same priority for queuing all calls.
- To add a test such as for working hours in the beginning, use a vector which ends with **`goto vector x unconditionally`**, where vector x has the loop with the announcing steps.
- If you have queue limiting, use a "goto step/vector y if calls-queued in skill 25 pri l > queue_limit" step ahead of step 2 to give the caller an alternate treatment if the call cannot be queued.

```
1. wait-time 0 secs hearing ringback
2. queue-to skill 25 pri l
3. announcement 1000  [All our specialists are busy handling other customers. Your call
is important to us, so please wait.]
4. goto step 6 if interflow-qpos <> 1
5. announcement 1001 [you are the next call to be answered]
6. goto step 8 if interflow-qpos <> 2
7. announcement 1002 [you have 1 call ahead of you]
8. goto step 10 if interflow-qpos <> 3
9. announcement 1003 [you have 2 calls ahead of you]
10. goto step 12 if interflow-qpos <> 4
11. announcement 1004 [you have 3 calls ahead of you]
12. goto step 14 if interflow-qpos <> 5
13. announcement 1005 [you have 4 calls ahead of you]
14. goto step 16 if interflow-qpos <> 6
15. announcement 1006 [you have 5 calls ahead of you]
16. goto step 18 if interflow-qpos <> 7
17. announcement 1007 [you have 6 calls ahead of you]
18. goto step 20 if interflow-qpos <> 8
19. announcement 1008 [you have 7 calls ahead of you]
20. goto step 22 if interflow-qpos <> 9
21. announcement 1008 [you have 8 calls ahead of you]
22. goto step 26 if interflow-qpos <= 9
23. announcement 1009 [There are more than 8 calls ahead of you]
24. collect 1 digits after announcement 3000 [If you would like to leave a callback
message dial 1 otherwise press # or
    continue to wait {10 sec timeout is the default but it is adjustable}]
25. goto vector 200 if digits = 1 [vector 200 provides callback messaging via the
messaging command and related treatment]
26. wait-time 60 secs hearing music {this is optional}
27. announcement 1000 [Our specialists are still busy, please continue to wait]
28. goto step 4 unconditionally
```

Resort reservation service

The example involves a resort company that places a variety of advertisements in magazines for information on a particular resort or state. A call center makes the reservations for the resort company. To satisfy the request of many callers, an effort is made to connect callers to an agent who has visited the resort that the callers are interested in visiting. The resort company has also decided to sell additional sightseeing packages if the agent has a regional accent.

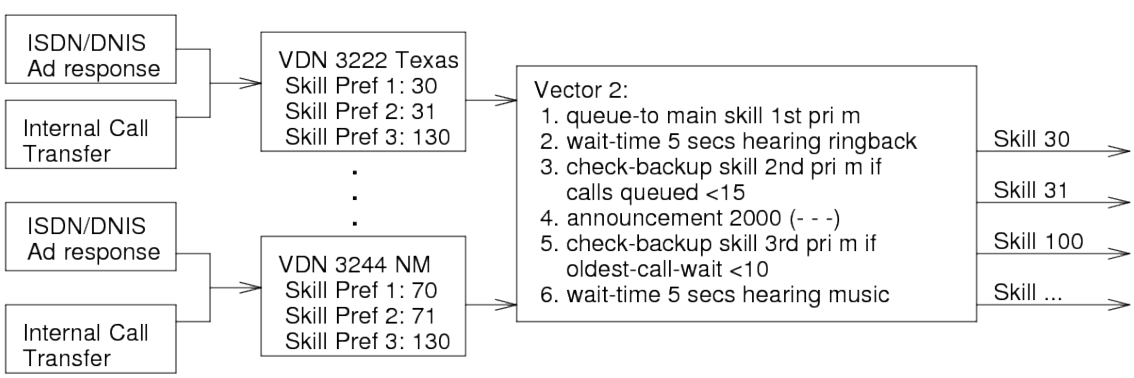
To respond to an advertisement, the caller can dial a number that directly routes the caller to a VDN for that state's resorts. As an alternative, the caller can dial the general number for the resort chain and be serviced using Call Prompting.

Specific number dialing

The call center is set up in such a way that a VDN with an accompanying set of VDN skill preferences is assigned to each state that has a resort. For example, the following table shows how skill preferences are assigned to the Texas VDN 3222.

Texas VDN 3222 skill preferences		
Skill preference	**Skill number**	**Agent skill**
1st	30	Agent who has a Texas accent and has visited resorts in Texas
2nd	31	Agent who has visited resorts in Texas
3rd	130	Any agent who can make the reservation

The following figure depicts how Call Vectoring processes a call to the VDN 3222.

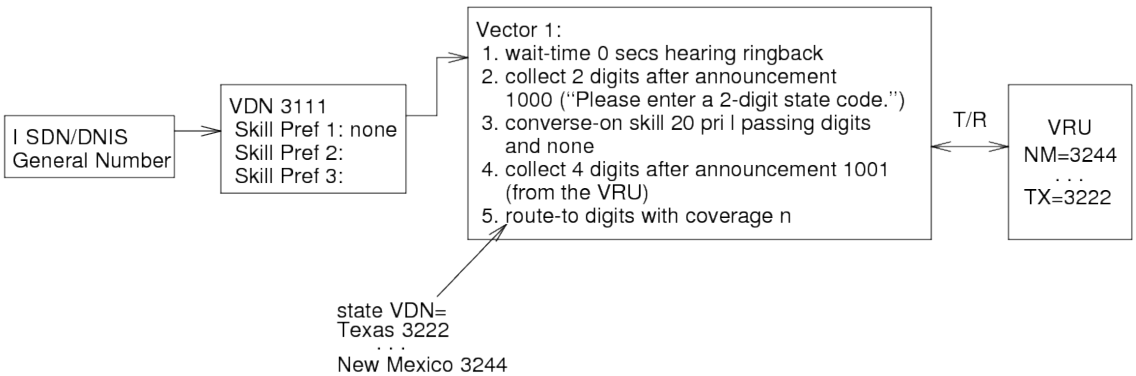


A single VDN for each state is assigned to vector 2. If a caller wants information on resorts in Texas and dials the correct number, for example, 615-3222, the call reaches Communication Manager and is directed to VDN 3222 which points to vector 2.

Once vector processing starts, the `queue-to skill` command in step 1 queues the call to the skill group that corresponds to the 1st VDN skill, 30-agent with a Texas accent who has visited resorts in Texas. If an agent with skill 30 is available, the agent answers the call. If the agent is not available, the `check skill` command in step 3 attempts to queue the call according to the stated conditions, if calls-queued < 15, to the skill group that corresponds to the 2nd VDN skill, 31-agent who has visited resorts in Texas. If step 3 fails, the **check skill** command in step 5 attempts to queue the call based on the stated conditions, that is, if the oldest-call waiting < 10, to the skill group that corresponds to the 3rd VDN skill, 100-any agent who can take a reservation.

General number dialing

Callers can dial a general number, for example, 615-3111. The caller is provided service in part using Call Prompting. The following figure depicts how a call to VDN 3111 is processed using Call Vectoring.



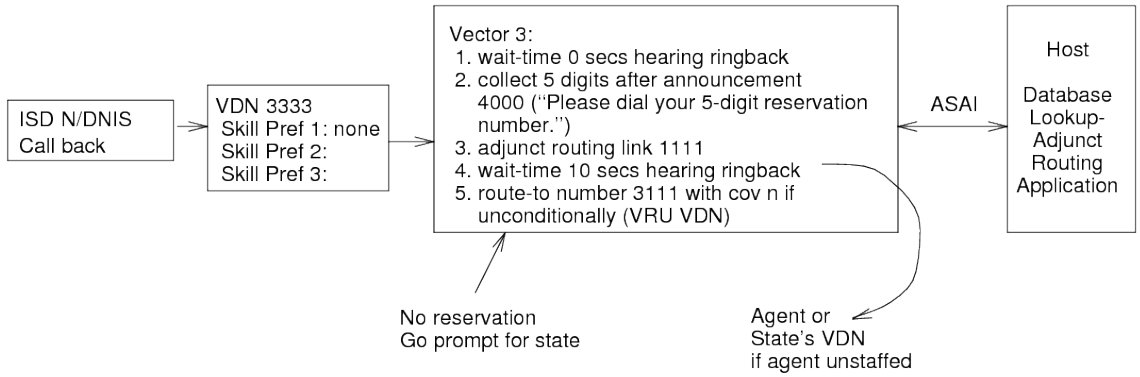
After the number is dialed, the call is directed to VDN 3111, which points to vector 1. Note that there are no skill preferences assigned to VDN 3111, which is the only VDN that is administered to point to vector 1. Therefore, VDN 3111 is used for calls from all states.

The **collect digits** command in step 2 of the previous vector first requests the caller to enter the appropriate 2-digit state code and then collects the digits. If the caller enters the correct code for Texas, which is 05, the **converse-on skill** command in step 3 delivers the call to the converse skill if there is a queue for the skill and the queue is not full, or if a VRU port is available.

When the VRU port responds, the step outpulses the state code 05 to the VRU using the passing digits parameter that is included in the command. Once the VRU receives this state code, the VRU in turn outpulses the Texas VDN (3222) to Communication Manager. Thereafter, the **collect digits** command in step 4 collects the digits that comprise this VDN. Finally, the **route-to digits** command in step 5 routes the call to Texas VDN 3222, which points to vector 2.

Callback provisions

After a caller makes a reservation for a resort site, the caller is given a callback number. Such a number is helpful if the caller requires more information or wants to check on some arrangement that was previously made. The following figure depicts one approach for activating callback provisions.



After the number is dialed, the call is directed to VDN 3333, which points to vector 3. Note that there are no skill preferences assigned to VDN 3333. Also, VDN 3333 is the only VDN that is administered to point to vector 3. Therefore, the VDN is used for calls from all states.

The **`collect digits`** command in step 2 of the previous vector first prompts the caller to enter the 5-digit reservation number and then collects the digits. Once the digits are collected, the **`adjunct routing link`** command, if successful, in step 3 causes Communication Manager to send the collected digits, along with other information, to the host in the ASAI adjunct routing request. The host then uses the digits to perform a database lookup for the agent who made the reservation and the resort that corresponds to the reservation. If the agent is currently logged in, the call is automatically routed to the agent. Information on the relevant reservation is displayed at the agent data terminal. If the agent is not logged in, the call is routed to step 5, where the **`route to`** command unconditionally routes the call to the VRU VDN 3111.

Attendant routing

The following example shows how Attendant Vectoring can be used to route calls in an attendant environment.

Note:

For the following vector examples, set **Tenant Partitioning** to `y`.

VDN 1999 Vector 1	VDN 2999 Vector 2	VDN 3999 Vector 3
1. wait-time 0 secs hearing ringback 2. goto step 7 if time-of-day is all 12:00 to 13:00 3. queue-to attd-group 4. goto step 8 if queue-fail 5. wait 999 secs hearing music 6. busy 7. route-to number 4000 with cov y if unconditionally 8. route-to number 93035381000 with cov y if unconditionally	1. wait-time 0 secs hearing ringback 2. queue-to attd-group 3. goto step 7 if queue-fail 4. announcement 9000 5. wait 999 seconds hearing music 6. disconnect after announcement 9001 7. queue-to hunt-group 1 8. goto step 11 if queue-fail 9.wait 999 secs hearing ringback 10. busy 11. route-to number 93035381000 with cov y if unconditionally	1. wait-time 0 secs hearing ringback 2. goto step 8 if time-of-day is all 12:00 to 13:00 3. queue-to attd-group 4. goto step 11 if queue-fail 5. announcement 9000 6. wait 15 seconds hearing music 7. goto step 4 if unconditionally 8. queue-to attendant 6000 9. goto step 11 if queue-fail 10. wait 999 secs hearing ringback 11. route-to number 93035381000 with cov y if unconditionally

Vector administration

- All stations are assigned TN 1 which is associated with attendant group 1, VDN 1999, and music source 1.
- All trunk groups are assigned TN 2 which is associated with attendant group 1, VDN 2999, and music source 2.
- All LDNs are assigned TN 3 which is associated with attendant group 2, VDN 3999, and music source 3.
- Extension 4000 is assigned to a hunt group 1.
- Extension 6000 is assigned to an attendant console for direct access.

Note:

Covered calls can be rerouted to an attendant group for a different tenant partition B when the call does not queue to the attendant group for tenant number A or for the out of hours case, and so on. The vector assigned to the attendant vectoring VDN for tenant A needs to include a failure branch to a “route-to ldn_number with cov y if unconditionally” step to route to the LDN extension of tenant B. The “with coverage” parameter of the route-to step must be set to “cov y” and the original coverage path that covers to the tenant A attendant vectoring VDN must have the “Cvg Enabled for VDN Route-to party?” field set to “y”.

Local attendant group access code

When a station dials the attendant access code, the call is redirected to vector 1. If it is lunch time, the call is sent to a hunt group and vector processing terminates. If it is not lunch time, the call is sent to attendant group 1. If an attendant is available, the call is directed to the attendant and vector processing terminates. Otherwise, the call is queued to the attendant group and the caller hears music from a music source that is assigned to TN 1 until an attendant answers the call. If the call cannot be queued, it is routed to a remote location with coverage, and vector processing terminates. If the call is unanswered after 999 seconds in the attendant queue, the caller hears a busy signal and vector processing terminates.

Note:

The **`route-to`** command leaves vector processing as soon as the call is successfully routed. So, if it is lunch time the call routes to the hunt group and all hunt group processing applies. If the group is assigned a queue, the call is queued. If the group is not assigned a queue and the coverage criteria is met, the call follows the hunt group's coverage path. If the hunt group is in night service, the call goes to the hunt group's night service destination. If the **`route-to`** command indicates coverage `n`, the hunt group's coverage path is not followed and vector step 7 applies.

Incoming trunk calls to attendant group

When a trunk that has the attendant group assigned as the incoming destination receives a call or when the attendant group addresses the call, the call is redirected to vector 2. The call is then sent to attendant group 1. If an attendant is available, the call is terminated to the attendant and vector processing terminates. Otherwise, the call is queued to the attendant group and the caller hears the announcement followed by music from the music source that is assigned to TN 2. If the call is unanswered after 999 seconds in the attendant queue, the caller is dropped after hearing an announcement and vector processing terminates. If queuing to the attendant fails, the call is queued to hunt group 1. If a member is available to take the call, the call is terminated to the member and vector processing terminates. If a member is not available and the call can be queued, the call is queued and the caller hears ringback until a member answers. If the call is unanswered after 999 seconds in the hunt group queue, the caller hears busy and vector processing terminates. If the call is not queued, the call is routed to the remote location and vector processing terminates.

Incoming LDN calls

When a call is received for a Listed Directory Number (LDN), the call is redirected to vector 3. If it is lunch time, the call is sent to attendant 6000. If the attendant is available, the call is answered and vector processing terminates. If the attendant is not available, the call is placed in queue and the caller hears ringback until the attendant answers the call. If the call is unanswered after 999 seconds in the attendant's queue, the call is sent to the remote location and vector processing terminates. If the call cannot be placed in the queue of attendant 6000, the call is routed to a remote location and vector processing terminates. If it is not lunch time, the call is sent to attendant group 2. If an attendant is available, the call is directed to the attendant and vector processing terminates. Otherwise, the call is queued to the attendant group and the caller hears an announcement followed by music from the music source assigned to Tenant Number (TN) 3 every 15 seconds. If the call cannot be queued, the call is sent to attendant 6000.

Note:

Vector 3 attempts to queue the call to attendant 6000. A **`route-to`** command can also be used, but ensure that an attendant is not assigned a coverage path. To change to a different tenant attendant group see note under Vector administration.

QSIG Centralized Attendant Service

This example shows how you can use Attendant Vectoring with Centralized Attendant Service (CAS).

CAS branch

If a call center wants to always play an announcement at a QSIG CAS branch before routing the call to the QSIG CAS main, an attendant VDN must be administered in the **QSIG CAS Number** field at the branch instead of the number to the QSIG CAS main attendant access code, which is 303-538-0 with an Automatic Alternate Routing (AAR) access code of 9, in this example. The following vector plays an announcement and then routes the call to the QSIG CAS main.

Administration for vector 1 of the attendant VDN is shown in the following Call Vector example.

```
change vector 1                                              Page 1 of 3
      CALL VECTOR

    Number: 1        Name: Night station service vector 4
Multimedia? n     Attendant Vectoring? y      Lock? y
     Basic? n  EAS? n  G3V4 Enhanced? n  ANI/II-Digits? n  ASAI Routing? n
 Prompting? y  LAI? n G3V4 Adv Route? n  CINFO? n  BSR? n   Holidays? n

01 announcement 9000
02 route-to number     93035380       with cov y if unconditionally
03
04
05
06
07
08
09
10
11
```

CAS main

Calls from a QSIG branch are sent to the QSIG CAS main with the main attendant access code as the destination address. Therefore, these calls automatically become attendant group calls. The VDN to which these calls are redirected depends on the Tenant Number (TN) of the incoming trunk.

Night station service

This example shows how to use Attendant Vectoring for night service.

Night station service vectors 4 and 5

```
change vector 4                                              Page 1 of 3
      CALL VECTOR

    Number: 4        Name: Night station service vector 4
Multimedia? n     Attendant Vectoring? y      Lock? y
     Basic? n  EAS? n  G3V4 Enhanced? n  ANI/II-Digits? n  ASAI Routing? n
 Prompting? y  LAI? n G3V4 Adv Route? n  CINFO? n  BSR? n   Holidays? n

01 route-to     number 9303538100       with cov y if unconditionally
02
03
04
05
06
07
08
09
10
11
_ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _

change vector 5                                              Page 1 of 3
      CALL VECTOR

    Number: 5        Name: Night station service vector 4
Multimedia? n     Attendant Vectoring? y      Lock? y
     Basic? n  EAS? n  G3V4 Enhanced? n  ANI/II-Digits? n  ASAI Routing? n
 Prompting? y  LAI? n G3V4 Adv Route? n  CINFO? n  BSR? n   Holidays? n

01 route-to     number 6000        with cov n if unconditionally
02 route-to     number 93035381000 with cov y if unconditionally
03
04
05
06
07
08
09
10
11
```

Administration for vector 4 and vector 5 of VDN 4999 is as follows.

- Trunk group 1 is assigned TN 2 which is associated with attendant group 1, and night destination 4999.
- Trunk group 2 is assigned TN 1 which is associated with attendant group 2, and night destination 5999.
- Extension 6000 is assigned to a station.
- System night service is on.

When a non-DID call comes in on trunk group 1, the call is redirected to VDN 4999 which routes it to a remote location.

When a non-DID call comes in on trunk group 2, the call is redirected to VDN 5999 which routes it to station 6000. If station 6000 is unavailable, the call does not cover on station 6000's coverage path. Vector processing continues and routes the call to a remote location.

Note:

When station night service is active, calls are processed according to the administered night destination for the trunk group, not the night destination for the associated TN. In other words, these are not attendant group calls. If the night destination is assigned as `attd` or left unassigned, the calls become attendant group calls and are processed according to the partitions night destination.

Holiday Vectoring

This is an example of a vector that directs calls to special processing due to a holiday or special event. Holiday Vectoring is an enhancement that simplifies vector writing for holidays.

In this example, a commercial bank is headquartered in Germany and has branches across Europe. The bank recently established a U.S. presence with branches in the New York City metropolitan area. The credit card division operates two 100-agent call centers in Ireland and Germany and one 50-agent call center in the U.S.

All agents in the call centers are bilingual and are assigned to splits that handle calls from both English and German customers. The New York call center is open 24 hours a day, often receiving calls routed from the Irish and German call centers, after the centers close at 6:00 p.m. local time.

Due to a large number of bank holidays in Europe, up to 30 days a year, Holiday Vectoring can be used to create vectors that distribute calls automatically on holidays. Holiday Vectoring saves cost and the time spent on writing vectors for date-related processing. This is turn saves business that is lost due to abandoned calls if vectors are not re-administered for holidays.

The following example indicates that, beginning on December 24 and continuing through 6:00 am on January 2, incoming calls to the call center in Germany will be processed as Christmas holiday calls.

Note:

Because date ranges must be within the same calendar year, New Year's day must be entered as a separate item.

Setting up a holiday table

```
change holiday-table 1                                     page 1 of 1
                                HOLIDAY TABLE

    Number: 1             Name: Bank Holidays

          START                  END
    Month Day Hour Min    Month Day Hour Min      Description
    12    24  00   00     12    31  23   59       Christmas
    01    01  00   00     01    02  06   00       New Year's Day
```

After setting up the Holiday Tables screen modify the vector processing for the holidays. On the Call Vector screen, enter the new goto conditional for the holidays.

When you set the **Holiday Vectoring** field to `y`, a field on the Call Vector screen identifies if the vector on which you are currently working is a holiday vectoring vector, as shown in the following example.

Modifying a vector to route according to a holiday table

```
change vector x                                                   page 1 of 3

                                      CALL VECTOR


    Number: xxx      Name: ___________________________
Multimedia? n      Attendant Vectoring? n    Meet-me Conf? n           Lock? y
     Basic? y   EAS? n   G3V4 Enhanced? y   ANI/II-Digits? y   ASAI Routing? y
 Prompting? y   LAI? y  G3V4 Adv Route? y   CINFO? y   BSR? y   Holidays? y
 Variables? y                3.0 Enhanced? y

01 ____________
02 ____________
03 ____________
04 ____________
05 ____________
06 ____________
07 ____________
08 ____________
09 ____________
10 ____________
11 ____________
```

The **Vectoring (Holiday)** field is a display-only field and appears only when you set the **Holiday Vectoring** field on the Customer Options screen to `y`. If either **Vectoring (Basic)** or **Attendant Vectoring** are set to `y`, the **Vectoring (Holiday)** field can be set to `y`.

```
 change vector 1                                                Page 1 of 3
                                  CALL VECTOR

    Number: 1        Name: In Germany
Multimedia? n      Attendant Vectoring? n    Meet-me Conf? n           Lock? y
     Basic? y   EAS? n   G3V4 Enhanced? n   ANI/II-Digits? n   ASAI Routing? n
 Prompting? y   LAI? n  G3V4 Adv Route? n   CINFO? n   BSR? n   Holidays? y

01 goto         vector 2  if holiday         in    table 1
02 route-to     number 123456789             with cov n if unconditionally
03
04
05
06
07
08
09
10
11
```

```
change vector 3                                                Page 1 of 3
                                  CALL VECTOR

    Number: 3        Name: In Ireland
Multimedia? n      Attendant Vectoring? n    Meet-me Conf? n           Lock? y
     Basic? y   EAS? n   G3V4 Enhanced? n   ANI/II-Digits? n   ASAI Routing? n
 Prompting? y   LAI? n  G3V4 Adv Route? n   CINFO? n   BSR? n   Holidays? y
```

```
01 goto          step   2  if holiday        in    table 2
02 route-to      number 45678          with cov n if unconditionally
03 stop
04 announcement 2721
05
06
07
08
09
10
11
```

The setup for the vector routes the call to a call center in the U.S. For example, if someone in Europe calls the bank before 6:00 a.m. on January 2, the call is routed to the U.S. call center. If someone in Europe calls after 6:00 a.m. on January 2, the call is routed to the German call center.

After you have assigned holiday tables to several vectors, you can use the **`list usage holiday-table`** command to view which vectors and vector steps are using the selected holiday table.

```
list usage holiday-table x
                              LIST USAGE REPORT

     Used By
     Vector                 Vector Number 1                     Step 1
     Vector                 Vector Number 3                     Step 1
```

Network Call Redirection

The example shows the primary, status poll, and interflow vectors that redirect calls on the public network using Network Call Redirection (NCR).

An e-Commerce company has three call centers. In an effort to reduce costs, the company implemented NCR to redirect calls on a public network and to reduce the trunking costs between the three call center offices. Best Service Routing (BSR) is implemented to increase the agent utilization.

The company receives calls from a public network. Trunks used to deliver calls from the public network have been assigned Network Call Transfer (NCT) capabilities. NCT occurs after the incoming call is initially answered. With NCT, Communication Manager is required to set up the second leg of the call and then to wait for the second site to acknowledge before requesting the public network to transfer the first leg of the call to the second leg, and before the public network drops the trunks to Communication Manager. The benefit is that Communication Manager retains control over the call and can redirect the call using the trunk-to-trunk method if the NCT invocation fails.

After the second leg of the call is initiated and acknowledged by the public switch, the public network joins the original caller to the redirected-to endpoint and drops both the original call and the second leg of the call at the redirecting Communication Manager.

To activate the NCR feature for each site, the Communication Manager administrator ensures that the **Net Redir** field on the BSR Application Table screen is set to `y` for the location entry.

The company has set up IP trunking to emulate ISDN PRI and uses this capability to poll remote sites for possible NCR.

The following sections provides examples on how to set vectors at each site, to use the public network with NCR, as opposed to IP trunking, to route a call from one site to another.

Primary vector for NCR example

A call arrives at the e-Commerce location 1 and a primary vector processes the call. The vector begins the BSR process by checking the specified resources. The following example shows the primary vector for incoming call processing at the e-Commerce location 1.

Primary vector

```
1. wait-time 0 secs hearing ringback
2. consider split 1 pri m  adjust-by 0
3. consider location 2     adjust-by 30
4. consider location 3     adjust by 10
5. queue-to-best
```

If location 2 returns the lowest EWT, the call is routed to location 2.

Status poll vector for NCR example

To collect information from a remote Communication Manager, the command **`consider`** *location 2* adjust-by *30* in the primary vector places a status poll using IP trunking to the status poll vector on Communication Manager at location 2.

Status poll vector

```
1. consider split 2   pri m   adjust-by 0
2. consider split 11  pri m   adjust-by 0
3. reply-best
```

The status poll returns the information to the origin Communication Manager. The call is not connected to the status poll VDN. Once the remote Communication Manager has returned the necessary information, the **`consider`** series in the primary vector at location 1 proceeds to the next vector step.

Interflow vector for NCR example

Communication Manager selects the site for call routing, and sends the call to the public network. The public network switch sets up a second leg of the call and passes the codeset 0 UUI information in the SETUP message, if this is supported. If you set the **Net Redir** field on the BSR Application screen to `y` for location 1, location 2, and location 3, Communication Manager sends a request to the public switch for call transfer over the public network.

For incoming 800 number calls from MCI DMS-250 network switches, you must ensure that the vector reached by the second leg call is answered immediately and an ISDN CONNect message is sent. This can be accomplished with a **`wait 0 secs hearing music`** or an **`announcement`** step as the first step in the receiving interflow vector.

BSR example of interflow vector on a remote Communication Manager

```
1. announcement    83345
2. consider split  2   pri m  adjust-by 0
```

```
3. consider split  11  pri m  adjust-by 0
4. queue-to best
```

The public network connects the second leg of the call to the second site and drops the trunk to Communication Manager.

Best Service Routing using Expected Wait Time and agent adjustments

A catalog company has three call centers, two in the United States and one in France. Best Service Routing (BSR) is implemented across the sites. The catalog company uses the UCD-MIA call distribution method at each site and uses the UCD-MIA available agent strategy for the VDN that is active for the call. The catalog company uses the adjust-by parameter in the **`consider`** vector step to select the best agent at any site.

The catalog company uses the adjust-by parameter for call delivery based on the adjusted idle time for the agents, so that a remote site is not selected when agent idle time difference is not significant.

To activate **BSR Available Agent Adjustments**, set the **Available Agent Adjustments for BSR** field on the Feature-Related System Parameters screen to `y`.

To use **BSR Available Agent Adjustments**, you must change the adjust-by value in the **`consider`** vector steps to include a percentage adjustment appropriate for each call center. In this example, adjust-by values are defined as `0` for the first call center, 20 percent for the second call center, and 20 percent for the third call center. The adjustment applies if there is an agent surplus at more than two call centers.

Note:

If the agent idle time is more than 100 seconds, the idle time is decreased by the assigned percentage. If the agent idle time is less than 100 seconds, the idle time is decreased by the adjustment in seconds.

The following table summarizes how the adjustment affects the idle times for each site.

Idle time adjustment calculations				
Location	**Agent idle time (secs)**	**Adjust by xx percentage**	**Calculation**	**Adjusted idle time (secs)**
Incoming split 1 at location 1	40	0 (Since the adjust-by parameter in the **`consider`** step is set to zero, no adjustment is made.)	0	40
Location 2	50	20	50 - 20	30

Table continues…

Idle time adjustment calculations				
Location	**Agent idle time (secs)**	**Adjust by xx percentage**	**Calculation**	**Adjusted idle time (secs)**
Location 3	100	20	100 - 20 (20 percent of 100)	80
The agent idle time at location 2 is less than 100 hence, the adjusted idle time is the difference between the two numbers. The agent idle time for location 3 is 100 hence, the adjusted idle time is reduced by 20 percent.				

Primary vector for BSR using EWT and agent adjustments example

An incoming call arrives at location 1 and the primary vector processes the call. The vector begins the BSR process by checking the specified resources. An example primary vector for incoming call processing at location 1 is shown in the following example.

Primary vector with adjustments

```
1. wait-time 0 secs hearing ringback
2. consider split    1  pri m   adjust-by 0
3. consider location 2          adjust-by 20
4. consider location 3          adjust-by 20
5. queue-to-best
```

In this example, the **`consider`** commands in steps 2, 3, and 4 collect information to compare local split 1 with location 2 and location 3. In each case, an available agent is found and an agent idle time returned. The adjust-by in steps 3 and 4 adjusts the value of the agent idle time. Step 5 queues the call to the best location found.

Status poll vector for BSR using EWT and agent adjustments example

To collect information from a remote Communication Manager, the **`consider location 2`** adjust-by `20` command in the primary vector places a call to the status poll vector on Communication Manager at location 2. An example of a status poll vector is indicated as follows:

Status poll vector

```
1. consider split 2   pri m  adjust-by 0
2. consider split 11  pri m  adjust-by 0
3. reply-best
```

The status poll returns the information to the first Communication Manager. The call does not connect to the status poll VDN.

The vector compares splits 2 and 11, identifies the best of the two splits, and sends the information back to the first Communication Manager with the **`reply-best`** command. You can use the adjust-by parameter on the remote Communication Manager to adjust the Expected Wait Time (EWT) or agent idle time returned by either of the splits. When you apply adjustments at the origin and remote Communication Manager, the two adjustments are added at the originating Communication Manager.

The **consider** command is ISDN-neutral and does not return answer supervision. The status poll call is dropped when the **reply-best** step executes, but the ISDN DISCONNect message returned to the first Communication Manager contains information from the best split at location 2. Once the remote Communication Manager returns the necessary information, the consider series in the primary vector on the first Communication Manager proceeds with the next vector step.

Interflow vector for BSR using EWT and agent adjustments example

Based on the values from the table, location 2 is the best site. The **queue-to best** command in the primary vector interflows the call to the interflow vector at location 2 as indicated in the following example.

Interflow vector on remote Communication Manager

```
1. consider split  2   pri m  adjust-by 0
2. consider split  11  pri m  adjust-by 0
3. queue-to best
```

The interflow vector checks the status of both splits to get the latest information and queues the call to the best split. Note that the **consider** sequences in the interflow vector and the status poll vector are identical except for the last step.

When the call is interflowed, the call is removed from the queue at the origin Communication Manager and any audible feedback at the origin Communication Manager is terminated.

Dial by Name

Use the Dial by Name feature to dial a number by entering the person’s name from your touchtone keypad. You can access this feature by using Call Vectoring and the integrated announcement circuit pack. This creates an auto-attendant procedure whereby callers can enter a person’s name instead of the extension number. The system processes characters of the name received, and when a match is found, the number is dialed automatically.

Note:

You must set the **Dial by Name** field to `y` to create a vector for this purpose.

A typical scenario includes the following call processing features:

- When a call comes in to the system (usually to an LDN call), a vector routes the call to an announcement that says, *Hello. You have reached A1 Hotel. Press 0 for an operator, press 1 for front desk, press 2 if you know the extension, press 3 if you know the name, press 4 to choose from a list of extensions, or press 5 to hear the options again.*
- If the caller selects 3, the caller is prompted to enter the person’s name.
- As soon as a match is found, the call is placed at the person's extension.

You can assign several vectors that define how to handle calls as users select the different prompts. The following example shows an auto-attendant procedure that can be used to access the Dial by Name feature. Step numbers 1-20 contain the basic auto-attendant steps, and steps 21-32 contain the Dial by Name steps.

Example Dial by Name vector

```
change vector 2                                              Page   1 of   3
                                   CALL VECTOR

    Number: 2                 Name: Dial by Name
                   Attendant Vectoring? y              Lock? n
     Basic? y   EAS? n    G3V4 Enhanced? n    ANI/II-Digits? n    ASAI Routing? n
 Prompting? y   LAI? n   G3V4 Adv Route? n    CINFO? n   BSR? n    Holidays? y

01 wait-time     2   secs hearing ringback
02 collect       1  digits after announcement 381
03
04 route-to      number 0                with cov n if digit              =  0
05 route-to      number 105              with cov n if digit              =  1
06 goto          step   12  if digits           =      2
07 goto          step   21  if digits           =      3
08 goto          step   19  if digits           =      4
09 goto          step   16  if digits           =      5
10 route-to      number 0                with cov n if unconditionally
11
_ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _

change vector 2                                         Page 2 of 3
                             CALL VECTOR

12 collect       3  digits after announcement 382
13 route-to      digits with coverage y
14 route-to      number 0               with cov n if unconditionally
15
16 goto          step   2   if unconditionally
17
18
19 collect       3  digits after announcement 383
20 goto          step   13  if unconditionally
21 collect       4 digits after announcement 661
22 route-to      name1 with coverage y
_ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _

change vector 2                                         Page 3 of 3
                             CALL VECTOR

23 goto          step 30 if nomatch
24 collect       11 digits after announcement 662
25 route-to      name2 with coverage y
26 goto          step 30 if nomatch
27 collect       2 digits after announcement 663
28 route-to      name3 with coverage y
29 goto          step 30 if nomatch
30 collect       1 digits after announcement 660
31 goto          step 21 if digits = 1
32 route-to      number 0                with cov n if unconditionally
```

This example includes the following call processing features and functionalities:

1. When someone calls the system, the caller receives a ringback for 2 seconds.
2. Announcement 381 plays. This announcement asks the caller to do one of the following:
    - Press 0 for an operator. If the caller presses 0 or waits for the time-out, the call is routed to the operator.

- Press 1 for front desk. If the caller presses 1, the call is routed to extension 105, which is the front desk.
- Press 2 if the caller knows the extension. If the caller presses 2, the call is routed to announcement 382, which prompts the caller to dial the extension.
- Press 3 if the caller knows the person's name. If the caller presses 3, the following sub-procedure occurs:
    a. Announcement 661 plays prompting that the caller enter the first four characters of the person's last name.
        - If there is a match, the call is redirected.
        - If there are multiple matches, continue with ii.
        - If there is no match, go to iiii.
    b. Announcement 662 plays prompting the caller to enter the rest of the person's last name, followed by the # key.
        - If there is a match, the call is redirected.
        - If there are multiple matches, continue with iii.
        - If there is no match, go to iv.
    c. Announcement 663 plays prompting the caller to enter the first two characters of the person's first name.
        - If there is a match, the call is redirected.
        - If there is no match, continue with iv.
    d. Since there are still no matches, announcement 660 plays prompting the caller to press 1 to try again, or press 0 to speak to an operator.
- Press 4 if the caller knows the department (such as housekeeping) that the caller wishes to access. If the caller presses 4, the call is routed to announcement 383, which gives the caller a list of several departments that the caller can dial directly.
- Press 5 to start again. If the caller presses 5, the caller hears announcement 381 that repeats all of the options.
- If the caller dials anything else, the call is routed to the operator.

Agent Identifier available in VDN Return Destination

You can use the VDN Return Destination (VRD) feature with Interactive Voice Response (IVR) systems and Voice Response Units (VRU) along with a post-call survey application. From Call Center Elite 7.1 onwards, the Agent Identifier available in VRD feature, adds the *Last Agent Identifier (ID)* to the information available to the IVR system. The agent identifier is made available in a new system-defined variable type in vectoring, and is sent to an IVR application performing a post-call survey, when a customer call is redirected by the VDN Return Destination (VRD) feature into vector processing. The *Agent Identifier* can be included in User-to-User Information (UUI) using existing vector commands before routing the call to an external IVR system. Alternatively, the *Agent Identifier* can be used as a **converse-on** vector command *operand* , for older VRU or IVR systems. By adding the agent identifier in UUI, post-call surveys can furnish reports details down to the agent level.

Definition of “Last” Agent

Agent in this context is the agent who has disconnected from the customer session. However, the customer session is retained and the VRD feature is processing the session. UUI can contain the agent identifier for only one agent. For example, in case of a call flow that has multiple transfers and multiple agents handling that customer call in a sequence then the UUI contains the agent identifier only for the last agent who spoke to the customer. In this case, the IVR/VRU system with a post-call survey application receives the agent information only for the last agent. You must define only one *agent* variable in the system. The *agent* type variable is read-only and cannot be set.

```
change variables                                               Page  18 of  39
                       VARIABLES FOR VECTORS

Var Description                 Type    Scope Length Start Assignment       VAC
..
LA  Last Agent [Call Survey]    agent   L
```

You can use the *agent* variable only in the following conditions:

- As *operand1* when setting an *asaiuui* vector variable in a VRD vector.
- As a **converse-on** vector command operand [*operand1* or *operand2*] in a VRD vector.

Example of setting an asaiuui variable in a VRD vector with the Last Agent Login ID

Define the *asaiuui* variable with a *Length* matching the longest agent login ID in the system.

```
change variables                                               Page  31 of  39
                       VARIABLES FOR VECTORS

Var Description                 Type    Scope Length Start Assignment       VAC
..
UA  UUI Last Agent ID [7 digit] asaiuui   L    7       14
```

You must set the *asaiuui* variable with the *Last Agent Login ID* in a VRD vector only. If you try to set the variable in a non-VRD vector step, the system generates the Vector Event 354, which is an “Assignment not allowed” event.

```
display vector 890                                     Page   1 of   6
                          CALL VECTOR

    Number: 890       Name: VEC-810-UUI-VRD-VDN  Background BSR Poll? n
..
01 set          UA     = LA     SEL   7
02 route-to     number 1118003          with cov n if unconditionally
03
```

If the *Last Agent Login ID* is 7 digits, for example 570-3367, vector step 1 in the example, sets bytes 14-20 of the UUI that is associated with this call, with the login ID. When the call is routed to the IVR system by vector step 2 the *Agent Login ID* is included in the UUI.

You must design the IVR system application such as, Experience Portal with Orchestration Designer to read the portion of the UUI, bytes 14-20, containing the agent identifier. Since UUI is flexible and can be used to convey multiple information fields, the location of the agent identifier within the UUI must match across the vectoring, where the agent identifier is written, and Experience Portal, where the agent identifier is read.

Varying Agent ID Lengths

If the *Last Agent Login ID* is 5-digits long, for example 12345, the system adds leading zeros “0” to the *Agent Login ID* when setting the 7 bytes of the UUI. Bytes 14-20 of the UUI will contain digits 0012345. The IVR system is responsible for handling the leading “0s”, if there are varying *Agent Login ID* lengths in the system.

If the *Last Agent Login ID* is 9-digits long, for example 123456789, the last 7 digits of the *Agent Login ID* is used when setting the 7 bytes of the UUI. Bytes 14-20 of the UUI will contain digits 3456789. No vector events are generated. The system administrator is responsible for ensuring that the *asaiuui* variable is defined and set with a length that matches the longest agent login ID in the system.

Example of using the Last Agent Login ID as a converse-on vector command operand

You must use the *Last Agent Login ID* only as a **converse-on** vector command operand in a VRD vector. If you try to use the *Last Agent Login ID* as a **converse-on** vector command operand in a non-VRD vector step, the system generates the Vector Event 38, which is a “Variable not allowed” event.

```
display vector 890                                          Page   1 of   6
                          CALL VECTOR

    Number: 890       Name: VEC-810-UUI-VRD-VDN  Background BSR Poll? n
..
01 converse-on  skill 801  pri l passing LA and none
02 ..
```

Step 1 in the VRD Vector results in the outpulsing of the *Last Agent Login ID* digits as a DTMF digit stream to the VRU or IVR connected by the **converse-on** command.

Varying Agent ID Lengths

The system handles varying Agent ID digit lengths and only outpulses those digits and exactly that number of digits.

Use of vectors in business scenarios

This section presents several typical business scenarios that involve telephone use. More than one vector is used in the examples to illustrate how to handle each scenario. The vectors presented here are suggested solutions. Individual call centers use their own unique requirements and budget in selecting and writing vectors.

Emergency and routine service

For emergency calls, write a vector that plays the following announcement: *We are aware of the power outage in the northeastern part of the city. The crew has been dispatched. If you are calling for another reason, please wait for an operator.*

For non emergency calls, write a vector to offer prompts to callers to speak with an agent.

Emergency and routine service suggested solution 1

Call Vectoring

```
 1. wait-time 0 seconds hearing ringback
 2. announcement 4100 [We are aware of the power outage in the northeastern part of the
city. Crews have been dispatched.
    If you are calling for other reasons, please hold for an operator.]
 3. wait-time 2 seconds hearing ringback
 4. goto step 10 if calls-queued in split 1 pri l > 20
 5. queue-to split 1 pri l
 6. wait-time 6 seconds hearing music
 7. announcement 4200 [We are sorry. All our operators are busy at the moment. Please
hold.]
 8. wait-time 10 seconds hearing music
 9. goto step 7 if unconditionally
10. disconnect after announcement 4200 [We are sorry. All our operators are busy at the
moment. Please call back later.]
```

In step 2 of the example vector, the **announcement** command plays an emergency information and prompts the caller to hold if the caller wants to speak with an operator on another matter. If the caller holds, the caller hears several seconds of a ringback provided by the **wait-time** command in step 3. Thereafter, the **goto step** command in step 4 checks if there are more than 20 calls queued in split 1. If so, a branch is made to step 10, where the **disconnect after announcement** command informs the caller that the call cannot be serviced at this time and then drops the call.

On the other hand, if 20 or fewer calls are queued to split 1, the call is queued to the split by the **queue-to split** command in step 5. Thereafter, unless the call is answered, feedback in the form of music is provided by step 6 and an announcement urging the caller to hold is provided by step 7. After another wait with music period is provided in step 8. The **goto step** command in step 9 branches back to the aforementioned please hold announcement in step 7. The resulting announcement-wait loop (steps 7 through 9) is then repeated until either an agent answers the call or the caller hangs up.

Emergency and routine service suggested solution 2

Call Vectoring and Call Prompting

```
VDN (extension=1030   name=Hub   vector=30)
Vector 30:
      1. wait-time 0 seconds hearing ringback
      2. collect 1 digits after announcement 3000 [“We are aware of the power outage in
the northeastern part of the city.
         Crews have been dispatched. If you are calling for other reasons, please press
1.”]
      3. route-to number 1031 with cov y if digit = 1
      4. announcement 3100 [“Invalid Entry. Please try again.”]
      5. goto step 2 if unconditionally

VDN (extension=1031   name=Service   vector=31)
Vector 31:
      1. announcement 4000 [“Please hold. We will try to connect you to an operator.”]
      2. wait-time 2 seconds hearing ringback
      3. goto step 9 if calls-queued in split 1 pri l > 20
      4. queue-to split 1 pri l
      5. wait-time 6 seconds hearing music
      6. announcement 4200 [“We are sorry. All our operators are busy at the moment.
Please hold.”]
      7. wait-time 10 seconds hearing music
      8. goto step 6 if unconditionally
```

```
      9. disconnect after announcement 4200 ["We are sorry. All our operators are busy
at the moment. Please call back later."]
```

Suggested solution 2 involves both Call Vectoring and Call Prompting. The solution also involves two vectors instead of just one vector. The announcement portion of the **collect digits after announcement** command in step 2 of vector 30 plays an emergency information, then prompts the caller to press 1 if calling for some other reason.

If the caller holds the line but enters the incorrect touch-tone digit, the **route-to number** command in step 3 attempts to route the call to VDN extension 1031 according to the entered digit. However, because a number other than 1 is entered, the call is not routed to the VDN extension. Instead, control is passed to step 4, where the **announcement** command informs the caller of the input error and prompts the caller to try again. Thereafter, the **goto step** command in step 5 unconditionally sends control back to step 2, where the **collect digits** command ultimately collects the digit that was entered by the caller. The digit-input loop, steps 2 through 5, continues for as long as the caller enters an incorrect digit.

If the caller correctly enters digit 1 as requested by the **collect digits** command in step 2, the **route-to number** command in step 3 sends control to the vector whose VDN extension is 1031, vector 31.

Late call treatment suggested solution

Call centers are staffed by union agents who work under a contract stipulating agents to be free at 5:00 p.m. However, some callers might call just before the closing time. Write a vector to inform such callers about the office hours.

Late call treatment

```
 1. goto step 15 if time-of-day is all 1700 to all 0800
 2. goto step 15 if time-of-day is fri 1700 to mon 0800
 3. goto step 16 if calls-queued in split 1 pri l > 20
 4. queue-to split 1 pri l
 5. goto step 10 if time-of-day is all 1645 to all 1700
 6. wait-time 20 seconds hearing ringback
 7. announcement 100 [We're sorry, all of our agents are busy. Please hold.]
 8. wait-time 998 seconds hearing music
 9. stop
10. announcement 200 [Please call back between 8:00 A.M. and 5:00 P.M., Monday through
Friday.]
11. wait-time 30 seconds hearing music
12. goto step 14 if time-of-day all 1700 to all 1710
13. goto step 11 if unconditionally
14. disconnect after announcement 300 [We are sorry, our office is now closed. Please
call back between 8:00 A.M. and 5:00 P.M.,
    Monday through Friday.]
15. disconnect after announcement 400 [We are sorry, our office is closed. Please call
back between 8:00 A.M. and 5:00 P.M.,
    Monday through Friday.]
16. disconnect after announcement 500 [We are sorry, we cannot service your call at
this time. Please call back between
    8:00 A.M. and 5:00 P.M., Monday through Friday.]
```

In the example vector, specific treatment is provided for calls that come into Communication Manager after working hours, during the weekend, or as the working day comes to a close.

The **goto step** command in step 1 checks if the call is placed during nonworking hours, during the week. If the call is received at this time, a branch is made to step 15, where the **disconnect**

after announcement command first informs the caller that the office is closed and then drops the call. If the call is not received at the time specified in Step 1, control is passed to step 2, where another **goto step** command checks if the call is received during weekend hours. If the call is received during weekend hours, a branch is made to step 15. If the call is not being placed at this time, control is passed to step 3.

The **goto step** command in step 3 checks for the number of calls in split 1. If more than 20 calls are queued to split 1, control is passed to step 16, where the **disconnect after announcement** command first informs the caller that the call cannot be serviced at this time and then disconnects the call. If 20 or fewer calls are queued to split 1, control is passed to step 4, where the **queue-to split command** queues the call to split 1.

Control is then passed to step 5, where the **goto step** command checks if the current time is between 4:45 p.m. and 5:00 p.m. inclusive (very close to, if not, closing time). If the current time does not fall within this clock range, the **wait-time** command in step 6 provides the caller with 20 seconds of ringback. Thereafter, the **announcement** command in step 7 plays an appropriate hold message, and the **wait** command in step 8 provides the caller with 998 seconds of music. Finally, the **stop** command in step 9 halts vector processing, and the call remains in queue until either the agent answers the call or the caller hangs up.

If the current time is 4:45 p.m. to 5:00 p.m. Step 5 executes a branch to step 10, where an appropriate late caller announcement plays. Thereafter, the **wait-time** command in step 11 provides the caller with 30 seconds of music. Control is then passed to step 12, where the **goto step** command checks if the time is currently any time between 5:00 p.m. and 5:10 p.m., inclusive. If so, control is passed to step 14, where the **disconnect after announcement** command first informs the caller that the office is now closed and then invites the caller to call back at the appropriate time before finally disconnecting the call.

If the time is currently not between 5:00 p.m. and 5:10 p.m. (inclusive), control is passed to step 13, where the **goto step** command branches back to the **wait-time** command in step 11. The resulting loop consisting of steps 11 through 13 is repeated for as long as the time is between 5:00 p.m. and 5:10 p.m. (inclusive), or until the caller hangs up. Once step 12 is executed a second after 5:10 P.M., control is passed to step 14 as described previously.

Messaging option

Procedure

1. Write a vector that does the following if the oldest call waiting is in queue for longer than 75 seconds:
   - Sends the call to the messaging system.
   - Plays the following a messaging announcement: `All our MegaSports agents are busy. Please leave your name and phone number.`
2. Plays 30 seconds of ringback for the caller.
3. After ringback, plays an announcement for the caller followed by music.

Result

Suggested solution

Messaging option

```
 1. goto step 8 if oldest-call-wait in split 50 pri l > 74
 2. goto step 8 if calls-queued in split 50 pri l > 20
 3. queue-to split 50 pri l
 4. wait-time 30 seconds hearing ringback
 5. announcement 1000 [“All of our MegaSports agents are busy. Please wait.”]
 6. wait-time 998 seconds hearing music
 7. stop
 8. announcement 2000 [“We are sorry, all our MegaSports agents are busy. Please leave
a message after the tone.
     Otherwise, please call back between 8:00 A.M. and 5:00 P.M, Monday through Friday.
Thank you.”]
 9. messaging split 20 for extension 4000
10. disconnect after announcement 2050 [“We are sorry, we are unable to take your
message at this time. Please call
    back between 8:00 A.M. and 5:00 P.M., Monday through Friday. Thank you.”]
```

The **goto step** command in step 1 of the example checks if the oldest call waiting in split 50 has been waiting for longer than 75 seconds. If so, control is passed to step 8, where the **announcement** command first informs the caller that all agents are busy and then requests the caller to either call back during working hours or leave a message for the agent. If the caller chooses to leave a message, the **messaging split** command in step 9 is executed. Upon execution of the **messaging split** command, an attempt is made to connect the caller to AUDIX so the caller can leave a recorded message. If the split queue is full, or if the AUDIX link is out of service, termination to AUDIX is unsuccessful, and vector processing continues at the next vector step. This step, as is the case here, usually contains an announcement which is played to inform the caller that all agents are busy. If the caller is successfully connected to AUDIX, vector processing terminates and a message can be left for the specified mailbox. 4000, in this case.

In step 1, if the oldest call waiting in split 50 has been waiting for fewer than 75 seconds, control is passed to step 2, where another **goto step** command checks the number of calls in split 50. If more than 20 calls are queued to split 50, control is passed to step 8. Thereafter, the procedure for the messaging option that is provided in the previous paragraph is implemented. If there are 20 or fewer calls waiting in split 50, control is passed to step 3, where the **queue-to split** command queues the call to the split.

Chapter 5: How to improve performance

Improved performance depends on the following basic principles:

- Minimize the number of vector steps to process a call.
- Do not use vector steps which have a substantial probability of failure such as the following:
  - Calls made outside of business hours.
  - Queue to groups with less than desirable resources or characteristics.

Inefficient looping wastes processing resources. For example, performance can be compromised when a vector loops through steps too often. This is especially true with long queue times.

The following are some looping examples with suggestions on how to maximize performance:

- Audible Feedback
- Look-Ahead Interflow
- Check

Examples other than looping are as follows:

- After Business Hours
- Look-Ahead Interflow

All looping examples in this section use only loops within a single vector. You must be aware of looping to other vectors through the use of vector chaining. The same principles can be extrapolated from the looping examples. Creating a flow diagram is often helpful for identifying looping errors.

In addition to the example vectors, tables rating the relative performance costs of specific vector commands are also included.

Note:

Test vectors for performance in addition to call flow.

Looping examples

Audible feedback examples

Note:

Evaluate the length of the wait period between repetitions of an announcement and increase the length. For optimum performance, add a second announcement after the initial announcement and repeat the second announcement less often.

In the following example, an announcement indicates all representatives as busy. Hold the announcement every 10 seconds as long as the call is in queue.

Example: 10-second announcement interval

```
1. queue-to split 1
2. announcement 2770  ["All representatives are busy. Please hold."]
3. wait-time 10 seconds hearing music
4. goto step 2 if unconditionally
5. stop
```

The following example repeats the announcement every 60 seconds, improving performance.

Example: 60-second announcement interval

```
1. queue-to split 1
2. announcement 2770    ["All representatives are busy. Please hold."]
3. wait-time 60 seconds hearing music
4. goto step 2 if unconditionally
5. stop
```

In the following example, a second announcement indicates all representatives as still being busy. Hold, in addition to the initial announcement, and repeat the second announcement less often, say after every 120 seconds, improving performance.

Example: Follow-up announcement

```
1. queue-to split 1
2. announcement 2770    ["All representatives are busy. Please hold."]
3. wait-time 120 seconds hearing music
4. announcement 2771    ["All representatives are still busy. Please continue to hold."]
5. goto step 3 if unconditionally
6. stop
```

The following table compares the relative processing cost of the three examples by looking at the approximate number of vector steps executed when processing the call. Let us say the first announcement is 3 seconds long and the second announcement is 4 seconds long. The approximate number of vector steps executed for an audible feedback is indicated in the following table.

Initial conditions	Example 10 seconds announcement interval	Example 60 seconds announcement interval	Example follow-up announcement
An agent is available in split 1	1	1	1
Queuing time of 5 minutes	70	15	9

When a call is queued for 5 minutes, the number of vector steps drops dramatically when the time between announcements is increased and drops even more when a second announcement is added and the time between announcements is increased again. When an agent in split 1 is immediately available to answer the call, there is no difference in the number of vector steps for the three examples.

Look-Ahead Interflow examples

Use the interflow-qpos conditional to achieve first in, first out (FIFO) or near-FIFO call processing. If you do not have the interflow-qpos conditional, add a wait period between successive LAI attempts and extend the waiting period.

The following example continuously attempts an LAI as long as the call is in queue or until a look-ahead attempt succeeds.

Example: continuous look ahead - no delay

```
1. queue-to split 1 pri l
2. announcement 3000
3. wait-time 20 seconds hearing music
4. route-to number 93035555555 cov n if unconditionally
5. goto step 4 if unconditionally
```

The following example adds a delay so that the LAI attempt occurs every 10 seconds.

Example: look ahead with a 10–second delay

```
1. queue-to split 1 pri l
2. announcement 3000
3. wait-time 20 seconds hearing music
4. route-to number 93035555555 cov n if unconditionally
5. wait-time 10 seconds hearing music
6. goto step 4 if unconditionally
```

The following example increases performance by increasing the delay between the LAI attempts to 30 seconds.

Example: look ahead with a 30–second delay

```
1. queue-to split 1 pri l
2. announcement 3000
3. wait-time 20 seconds hearing music
4. route-to number 93035555555 cov n if unconditionally
5. wait-time 30 seconds hearing music
6. goto step 4 if unconditionally
```

The following table compares the relative processing cost of the three examples by looking at the approximate number of vector steps executed when processing the call with an announcement that is 5 seconds long.

Table 1: Approximate number of vector steps executed for look-ahead interflow examples

Initial conditions	Example look ahead with no delay	Example look ahead with a 10–second delay	Example look ahead with a 30–second delay
An agent is available in split 1	1	1	1
Queuing time of 5 minutes	up to 1,000	85	30

Check examples

When using the **check** command to queue a call to backup splits, ensure that an adequate time has elapsed before checking the backup splits again.

Note:

With EWT, the programming style used in this example is not optimal. The best approach is to use EWT to locate an appropriate split for the call and queue the call.

Continuous check

The following example checks the backup splits continuously as long as the call is in queue.

```
1. queue-to split 1 pri h
2. announcement 3000
3. wait-time 10 seconds hearing music
4. check split 21 pri m if available-agents > 0
5. check split 22 pri m if available-agents > 0
6. check split 23 pri m if available-agents > 0
7. check split 24 pri m if available-agents > 0
8. check split 25 pri m if available-agents > 0
9. goto step 4 if unconditionally
```

Check with 10 second delay

The following example adds a delay of 10 seconds to ensure that some time has elapsed before checking the backup splits again.

```
1. queue-to split 1 pri h
2. announcement 3000
3. wait-time 30 seconds hearing music
4. check split 21 pri m if available-agents > 0
5. check split 22 pri m if available-agents > 0
6. check split 23 pri m if available-agents > 0
7. check split 24 pri m if available-agents > 0
8. check split 25 pri m if available-agents > 0
9. wait-time 10 seconds hearing music
10. goto step 4 if unconditionally
```

Agent availability status cannot change every 10 seconds.

Check with 30 second delay

The following example adds a delay of 30 seconds to ensure that some time has elapsed before checking the backup splits again.

```
1. queue-to split 1 pri h
2. announcement 3000
3. wait-time 30 seconds hearing music
4. check split 21 pri m if available-agents > 0
5. check split 22 pri m if available-agents > 0
6. check split 23 pri m if available-agents > 0
7. check split 24 pri m if available-agents > 0
8. check split 25 pri m if available-agents > 0
9. wait-time 30 seconds hearing music
10. goto step 4 if unconditionally
```

The following table compares the relative processing cost of the three examples by looking at the approximate number of vector steps executed while processing the call. Assumption is that the announcement is 5 seconds long.

Table 2: Approximate number of vector steps executed for check examples

Initial conditions	Example continuous check	Example check with 10-second delay	Example check with 30-second delay
An agent is available in split 1	1	1	1
Queuing time of 5 minutes	up to 1,000	190	65

If a call is queued for 5 minutes, the number of vector steps drop dramatically under the following two conditions:

- when a delay is added before checking the backup splits again
- when the length of the delay is increased again

When an agent in split 1 is immediately available to answer the call, there is no difference in the number of vector steps for the three examples.

After business hours example

Recommendation: Test to see if the destination resources are available (such as during business hours) before queuing.

The following example queues calls to a hunt group regardless of the time of the call. When calls reach an office after business hours, the announcement is repeated until the caller hangs up.

Unconditional queuing to hunt group

```
1. queue-to split 1
2. announcement 5000  (“All agents are busy. Please hold.”)
3. wait-time 120 seconds hearing music
4. announcement 5001   (“All agents are still busy. Please continue to hold.”)
5. goto step 3 if unconditionally
```

The next example tests for business hours before queuing the call. If the call reaches the office after business hours, an announcement informs the caller of the business hours and the call is terminated.

Queue to hunt group with time-of-day conditional

```
1. goto step 7 if time-of-day is all 17:00 to all 8:00
2. queue-to split 1
3. announcement 5000  (“All agents are busy. Please hold.”)
4. wait-time 120 seconds hearing music
5. announcement 5001  (“All agents are still busy. Please continue to hold.”)
6. goto step 4 if unconditionally
7. disconnect after announcement 5001  (“Business hours are 8:00 AM to 5:00 PM, Please
call back then.”)
```

In the first example, unnecessary processing occurs when a call queues after business hours and the call is terminated only when the caller hangs up. As indicated in the second example, it is more economical to test for business hours before queuing a call.

Look-Ahead Interflow example

Note:

When using LAI, check if the receiving office is open for business.

Scenario: The sending Communication Manager is in Los Angeles with office hours from 8:00 AM to 5:00 PM PST and the receiving Communication Manager is in New York with office hours from 8:00 AM to 5:00 PM EST (05:00-14:00 PST). There is a three hour difference between the two locations.

The following example routes calls to the New York Communication Manager.

Unconditional LAI

```
1. queue-to split 1
2. route-to number 99145555555 cov n if unconditionally
3. announcement 2770      (All agents are busy. Please hold.)
4. wait-time 120 seconds hearing music
5. goto step 3 if unconditionally
6. stop
```

The next example tests first to see if the New York Communication Manager is open before requesting a queue to the New York Communication Manager, preventing unnecessary processing.

LAI with Time-of-Day (TOD) condition

```
1. queue-to split 1
2. goto step 4 if time-of-day is all 14:00 to all 05:00
3. route-to number 99145555555 cov n if unconditionally
4. announcement 2770        (All agents are busy. Please hold.)
5. wait-time 120 seconds hearing music
6. goto step 4 if unconditionally
7. stop
```

Refer to the next example if you enabled **Advanced Routing**. In this case, the Expected Wait Time feature can be used to determine if placing an LAI call attempt is useful.

LAI with Expected Wait Time (EWT) and TOD conditions

```
1. queue-to split 1
2. goto step 5 if expected-wait for call < 30
3. goto step 5 if time-of-day is all 14:00 to all 05:00
4. route-to number 99145555555 cov n if unconditionally
5. announcement 2770        (All agents are busy. Please hold.)
6. wait-time 120 seconds hearing music
7. goto step 5 if unconditionally
8. stop
```

In the examples, note that there is no reason to attempt an interflow if the call is answered quickly at the main Communication Manager.

Chapter 6: Security issues

Call Vectoring can be integrated into the security of your switch. For example, Call Vectoring and Call Prompting can be used to help prevent unauthorized users from gaining access to the switch using the Remote Access feature. This section explains how this is done.

Remote access

Abuse of remote access on the switch is one of the main methods by which unauthorized users obtain telephone services illegally. This section explains how a number of Call Vectoring features can be used to prevent unauthorized use of the remote access feature. No new development is required for any of these services.

Two methods are available:

- Front-ending remote access
- Replacing remote access

Front-ending remote access

This method gives authorized external callers a VDN extension to call instead of the remote access extension, which is kept private. The corresponding call vector can then implement a number of security checks before routing callers to the remote access extension. Routing can be done using a **`route-to number`** or **`route-to digits`** step.

The following advantages are possible using this method:

- Call Vectoring can introduce a delay before the dial-tone is provided to the caller. Immediate dial-tone is often one criterion searched for by a hacker's programs when the hacker is trying to break into a system.
- A recorded announcement declaring that the use of Communication Manager services by unauthorized callers is illegal and that the call is subject to monitoring or recording can be played for the caller.
- Call Prompting can be used to prompt for a password. In such a case, the call is routed only if there is a match on the password.
- Use of the remote access extension can be limited to certain times of the day or certain days of the week.

- Real-time and historical reports on the use of the remote access feature can be accessed from CMS or from BCMS.
- Different passwords can be used on different days of the week or at different times during the day.
- Many VDNs that call the remote access extension can be identified. Accordingly, individuals or groups can be given their own VDN with unique passwords, permissions and reports. Any abuse of the system or security leak can then be attributed to an individual or a group.
- The caller can be routed to a VRU using the `converse-on` step where more sophisticated security checking, such as speaker recognition, can take place.
- Anyone failing any of the security checks can be routed to a security VDN that routes the caller to security personnel with a display set or to a VRU. Such a call shows the attempted password on the display. If the call is passed to a VRU, the VDN, the ANI or the prompted digits can be captured. CMS and BCMS reports on this security violation VDN gives information on how often and when security violations occur.

Replacing remote access

For this method, the remote access extension is not used. More than one VDN is designed to access call vectors that can employ all of the security checks described in the previous section. The same reports and monitoring or recording capabilities described in the previous section can also be used. Instead of routing to the remote access extension, the vector collects digits from the caller and routes to the given destination if there is a match on the password.

Again, multiple VDNs can be created for individuals or groups with different security checks and different permissions or restrictions. Destination numbers provided by callers can be screened by the vectors and denied if the user does not have permission to access that destination. For example, an individual user can be restricted from placing calls to numbers beginning with area codes 303 and 908.

EAS

With EAS, agent stations can be locked when not staffed. This is accomplished by assigning the station a COR that does not allow outbound calls or restricts outbound calls from toll calls.

EAS agents have an optional password of up to nine digits to log in. This password is not displayed on DCP terminals when the agent is entering the password on the dial pad.

Limiting outside access using VDN COR restrictions

Routing calls through Communication Manager with Call Vectoring can raise some security issues. A VDN has a COR. Calls processed by the vector carry the permissions and restrictions associated with the COR of the VDN.

For example, if a vector in Communication Manager is written to collect digits and then to route to the digits dialed, the restrictions on what calls can be placed are determined by the COR of the latest VDN. Also, checks can be made on the digits that are dialed, using **`goto if digits`** vector commands. For example, **`goto if digits = 123`** to disallow routing to undesired destinations. The **`collect digits`** step can also be limited to collect only the number of digits required, for example, only collecting five digits for internal dialing.

An incoming caller can access Trunk Access Codes, some FACs, and most other sets of dialed digits. To deny incoming callers access to outgoing facility paths, configure the COR of the VDN to deny outgoing access. The configuration must include the following:

- Lowering the Facility Restriction Level (FRL) in the COR to the lowest acceptable value. FRL=0 provides the most restricted access to network routing preferences.
- Assigning a Calling Party Restriction of Toll or Outward denying Facility Test Call capability.
- Blocking access to specific CORs assigned to outgoing trunk groups using the Calling Permissions section of the Class Of Restriction screen.

Review the CORs assigned to the VDNs. If not restricted, assign restrictions on the VDN or use goto tests on the digits to prevent callers from exiting the system using the vector.

Meet-me Conference security issues

Meet-me Conference is a potential security problem. If Meet-me Conference VDNs are assigned without access codes, hackers can tie up Meet-me Conference facilities, keeping others from conducting legitimate business and can potentially access the switch and use the switch to make unauthorized calls. Therefore, all Meet-me Conference VDNs must have access codes known to the users only on a need-to-know basis. You must also change the access codes on a regular basis to reduce the risk of unauthorized access.

If a user tries to change the access code of a Meet-me Conference and is unsuccessful, or if a user tries to access a Meet-me Conference and uses an invalid access code, a meet-me event is logged. For more information, see “Tracking unexpected events” in the *Programming Call Vectoring Features in Avaya Aura® Call Center Elite* document.

Vector-initiated service observing

The following restrictions can be used with vector-initiated service observing to guard against unauthorized use.

- Call prompting commands can be used in service observing vectors to provide passcode protection, and to limit access to observing specific destinations or verified caller entered digits.

- Time of Day/Day of Week checks can be incorporated in service observing vectors.
- A vector can be created to be used exclusively for service observing.
- For a VDN to be observed as the result of a route-to command, the VDN must have a COR that allows it to be observed.
- The calling permissions of the COR assigned to the service observing VDN in conjunction with the can be observed settings of the COR assigned to the destination determine what agents, stations, or VDNS can be observed.

Voice Response Integration (VRI)

When a **`converse`** step is used to access a VRU application that returns data for a collect digits step, the opportunity for toll fraud exists when the VRU application fails to return any data. Ensure that one of the following is true to prevent toll fraud:

- If the collected digits are used to route calls internally, be certain that the COR for the VDN does not allow calls to route externally.
- If you have to use collected digits to route calls externally, use a password to verify that the collected digits have been passed by the VRU application. In the following vector example the VRU application returns a three-digit password followed by the eight-digit external number. The vector routes calls without the correct password to a different vector and routes calls with the correct password to the collected digits.

VRI security

```
converse-on split 10 pri m passing none and none
collect 3 digits after announcement none
goto vector 23 if digits <> 234
collect 8 digits after announcement none
route-to digits with coverage n
```

Attendant Vectoring security issues

Security Violation Notification (SVN) referral calls can be directed to an attendant group. These are priority calls and, as such, cannot terminate to a VDN. However, when the calls are sent to the attendant group, the calls are treated as ordinary calls. Priority does not apply to attendant group processing. So, the calls are treated as normal attendant group calls and are sent through vector processing.

Chapter 7: Resources

Documentation

Title	Use the document to:	Audience
Supporting		
Avaya Aura® Communication Manager System Capacities Table	Read about the system capacity and system scalability.	Implementation engineers, sales engineers, and solution architects
Programming Call Vectoring Features in Avaya Aura® Call Center Elite	Write and edit call vectors.	Implementation engineers and system administrators
Avaya Aura® Call Center Elite Call Vectoring Feature Description	Learn how the Call Vectoring feature work and get to now details about feature characteristics, capabilities, capacities, and interactions.	Implementation engineers and system administrators
Understanding		
Avaya Aura® Call Center Elite Feature Reference	Read about Automatic Call Distribution (ACD) and Call Vectoring features.	All users of Avaya Aura® Call Center Elite
Avaya Aura® Communication Manager Feature Description and Implementation	Read about Avaya Aura® Communication Manager features.	All users of Communication Manager
Using		
Using Avaya Business Advocate	Learn how to use Business Advocate for agent selection and call selection.	Contact center managers, system administrators, and supervisors
Using Avaya 96X1 SIP Agent Deskphones with Avaya Aura® Call Center Elite	Know the prerequisites for using Avaya 96X1 SIP agent deskphones with Call Center Elite.	Implementation engineers and system administrators
Administering		
Administering Avaya Aura® Call Center Elite	Administer all the Call Center Elite features.	System administrators
Troubleshooting		

Table continues…

Title	Use the document to:	Audience
Troubleshooting Avaya Aura® Call Center Elite	Know how to troubleshoot common problems and denial events related to Call Center Elite.	All users who perform troubleshooting tasks in Call Center Elite
Implementation		
Planning for an Avaya Aura® Call Center Elite Implementation	Know how to transition from a basic call center environment to an Expert Agent Selection (EAS) and a Call Vectoring environment.	All users who perform Call Center Elite site preparation and planning tasks, including implementation engineers and sales engineers.

Finding documents on the Avaya Support website

Procedure

1. Go to https://support.avaya.com.
2. At the top of the screen, click **Sign In**.
3. Type your **EMAIL ADDRESS** and click **Next**.
4. Enter your **PASSWORD** and click **Sign On**.
5. Click **Product Documents**.
6. Click **Search Product** and type the product name.
7. Select the **Select Content Type** from the drop-down list
8. In **Select Release**, select the appropriate release number.

   For example, for user guides, click **User Guides** in the **Content Type** filter. The list only displays the documents for the selected category.
9. Press **Enter**.

Accessing the port matrix document

Procedure

1. Go to https://support.avaya.com.
2. At the top of the screen, click **Sign In**.
3. Type your **EMAIL ADDRESS** and click **Next**.
4. Enter your **PASSWORD** and click **Sign On**.
5. Click **Product Documents**.

6. Click **Search Product** and type the product name.
7. Select the **Select Content Type** from the drop-down list
8. In **Choose Release**, select the required release number.
9. In the **Content Type** filter, select one or both the following categories:
    - **Application & Technical Notes**
    - **Design, Development & System Mgt**

    The list displays the product-specific Port Matrix document.
10. Press **Enter**.

Avaya Documentation Center navigation

For some programs, the latest customer documentation is now available on the Avaya Documentation Center website at https://documentation.avaya.com.

Important:

For documents that are not available on Avaya Documentation Center, click **More Sites** > **Support** on the top menu to open https://support.avaya.com.

Using the Avaya Documentation Center, you can:

- Search for keywords.

  To filter by product, click **Filters** and select a product.
- Search for documents.

  From **Products & Solutions**, select a solution category and product, and then select the appropriate document from the list.
- Sort documents on the search results page.
- Click **Languages** ( ) to change the display language and view localized documents.
- Publish a PDF of the current section in a document, the section and its subsections, or the entire document.
- Add content to your collection using **My Docs** ( ).

  Navigate to the **Manage Content** > **My Docs** menu, and do any of the following:
  - Create, rename, and delete a collection.
  - Add topics from various documents to a collection.
  - Save a PDF of the selected content in a collection and download it to your computer.
  - Share content in a collection with others through email.

  - Receive collection that others have shared with you.

- Add yourself as a watcher using the **Watch** icon ( ).

  Navigate to the **Manage Content** > **Watchlist** menu, and do the following:

  - Enable **Include in email notification** to receive email alerts.
  - Unwatch selected content, all content in a document, or all content on the Watch list page.

  As a watcher, you are notified when content is updated or deleted from a document, or the document is removed from the website.

- Share a section on social media platforms, such as Facebook, LinkedIn, and Twitter.
- Send feedback on a section and rate the content.

Note:

Some functionality is only available when you log in to the website. The available functionality depends on your role.

Training

The following courses are available on www.avaya-learning.com. Enter the course code in the **Search** field, and click **Go** to search for the course.

Course code	Course title
ACIS-7391	
73600V	Implementing Avaya Aura® Call Center Elite 40 hours
7391X	Avaya Aura® Call Center Elite and Avaya Aura® Call Center Elite Multichannel Implementation Exam 1.50 hours
ACSS-7491	
74600V	Supporting Avaya Aura® Call Center Elite 16 hours
7491X	Avaya Aura® Call Center Elite and Avaya Aura® Call Center Elite Multichannel Support Exam 1.50 hours
2416W	Avaya Aura® Call Center Elite Fundamentals 0.5 hour for all audiences
2412W	Using Avaya Workspaces for Call Center Elite – Agents 0.5 hour for end-users
2414W	Using Avaya Workspaces for Call Center Elite – Supervisors 0.5 hour for end-users

Viewing Avaya Mentor videos

Avaya Mentor videos provide technical content on how to install, configure, and troubleshoot Avaya products.

About this task

Videos are available on the Avaya Support website, listed under the video document type, and on the Avaya-run channel on YouTube.

- To find videos on the Avaya Support website, go to https://support.avaya.com/ and do one of the following:
  - In **Search**, type `Avaya Mentor Videos`, click **Clear All** and select **Video** in the **Content Type**.
  - In **Search**, type the product name. On the Search Results page, click **Clear All** and select **Video** in the **Content Type**.

    The **Video** content type is displayed only when videos are available for that product.

  In the right pane, the page displays a list of available videos.
- To find the Avaya Mentor videos on YouTube, go to www.youtube.com/AvayaMentor and do one of the following:
  - Enter a keyword or keywords in the **Search Channel** to search for a specific product or topic.
  - Scroll down Playlists, and click a topic name to see the list of videos available. For example, Contact Centers.

  Note:

  Videos are not available for all products.

Support

Go to the Avaya Support website at https://support.avaya.com for the most up-to-date documentation, product notices, and knowledge articles. You can also search for release notes, downloads, and resolutions to issues. Use the online service request system to create a service request. Chat with live agents to get answers to questions, or request an agent to connect you to a support team if an issue requires additional expertise.

Related links

Using the Avaya InSite Knowledge Base on page 306

Using the Avaya InSite Knowledge Base

The Avaya InSite Knowledge Base is a web-based search engine that provides:

- Up-to-date troubleshooting procedures and technical tips
- Information about service packs
- Access to customer and technical documentation
- Information about training and certification programs

- Links to other pertinent information

If you are an authorized Avaya Partner or a current Avaya customer with a support contract, you can access the Knowledge Base without extra cost. You must have a login account and a valid Sold-To number.

Use the Avaya InSite Knowledge Base for any potential solutions to problems.

1. Go to https://support.avaya.com.
2. At the top of the screen, click **Sign In**.
3. Type your **EMAIL ADDRESS** and click **Next**.
4. Enter your **PASSWORD** and click **Sign On**.

   The system displays the Avaya Support page.
5. Click **Support by Product** > **Product-specific Support**.
6. In **Enter Product Name**, enter the product, and press `Enter`.
7. Select the product from the list, and select a release.
8. Click the **Technical Solutions** tab to see articles.
9. Select **Related Information**.

Related links

Support on page 306

Index

D

E

F

H

I

S

T

U

V

W