Avaya™ Communication Manager Contact Center

Call Vectoring Guide for Business Communications System (BCS) and GuestWorks

555-233-518
Issue 1.0
May 2003

© 2003, Avaya Inc.
All Rights Reserved

Notice

Every effort was made to ensure that the information in this document was complete and accurate at the time of printing. However, information is subject to change.

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 working on your company's behalf). Be aware that there may 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 Fraud Intervention

If you *suspect that you are being victimized* by toll fraud and you need technical assistance or support, call Technical Service Center Toll Fraud Intervention Hotline at +1 800 643 2353 for the United States and Canada. For additional support telephone numbers, see the Avaya Web site:

http://www.avaya.com

Select **Support**, then select **Escalation Lists US and International**. This Web site includes telephone numbers for escalation within the United States. For escalation telephone numbers outside the United States, click on **Global Escalation List**.

Providing Telecommunications Security

Telecommunications security (of voice, data, and/or video communications) is the prevention of any type of intrusion to (that is, either unauthorized or malicious access to or use of) your company's telecommunications equipment by some party.

Your company's “telecommunications equipment” includes both this Avaya product and any other voice/data/video equipment that could be accessed via this Avaya product (that is, “networked equipment”).

An “outside party” is anyone who is not a corporate employee, agent, subcontractor, or working on your company's behalf. Whereas, a “malicious party” is anyone (including someone who may be otherwise authorized) who accesses your telecommunications equipment with either malicious or mischievous intent.

Such intrusions may be either to/through synchronous (time-multiplexed and/or circuit-based) or asynchronous (character-, message-, or packet-based) equipment or interfaces for reasons of:

- Utilization (of capabilities special to the accessed equipment)
- Theft (such as, of intellectual property, financial assets, or toll-facility access)
- Eavesdropping (privacy invasions to humans)
- Mischief (troubling, but apparently innocuous, tampering)
- Harm (such as harmful tampering, data loss or alteration, regardless of motive or intent)

Be aware that there may be a risk of unauthorized intrusions associated with your system and/or its networked equipment. Also realize that, if such an intrusion should occur, it could result in a variety of losses to your company (including but not limited to, human/data privacy, intellectual property, material assets, financial resources, labor costs, and/or legal costs).

Your Responsibility for Your Company’s Telecommunications Security

The final responsibility for securing both this system and its networked equipment rests with you - an Avaya customer's system administrator, your telecommunications peers, and your managers. Base the fulfillment of your responsibility on acquired knowledge and resources from a variety of sources including but not limited to:

- Installation documents
- System administration documents
- Security documents
- Hardware-/software-based security tools
- Shared information between you and your peers
- Telecommunications security experts

To prevent intrusions to your telecommunications equipment, you and your peers should carefully program and configure:

- your Avaya-provided telecommunications systems and their interfaces
- your Avaya-provided software applications, as well as their underlying hardware/software platforms and interfaces
- any other equipment networked to your Avaya products.

Trademarks

Alcatel is a trademark of Compagenie Financiere Alcatel, Paris, France.

AUDIX, BCMSVu, Callmaster, Conversant, DEFINITY, and GuestWorks are registered trademarks of Avaya, Inc.

GTX is a trademark of the Global TeleExchange.

MCI is a registered trademark of MCI WorldCom.

Avaya and MultiVantageare trademarks of Avaya, Inc.

Ordering Information

Call: Avaya Publications Center
Voice +1 800 457 1235
Fax +1 800 457 1764
International Voice +1 207 866 6701
International Fax+1 207 626 7269

Write: Globalware Solutions
200 Ward Hill Avenue
Haverhill, MA 01835 USA
Attention: Avaya Account Manager

Web: http://www.avayadocs.com

E-mail: totalware@gwsmail.com

Order: Document No. 555-233-518, Issue 1.0
May 2003

Avaya Support

Avaya provides a telephone number for you to use to report problems or to ask questions about your contact center. The support telephone number is 1-800-242-2121 in the United States. For additional support telephone numbers, see the Avaya Web site:

http://www.avaya.com

Select **Support**, then select **Escalation Lists US and International**. This Web site includes telephone numbers for escalation within the United States. For escalation telephone numbers outside the United States, click on **Global Escalation List**.

Acknowledgment

This document was written by the CRM Information Development group.

Avaya Communication Manager
Call Center Software
Call Vectoring Guide for
Business Communications System (BCS)
and Guestworks

Contents

About this document

Chapter 1: Call Vectoring overview

Chapter 2: Creating and editing call vectors

Chapter 3: Call Vectoring fundamentals

Chapter 4: Basic Call Vectoring

Chapter 5: Call Prompting

Chapter 6: Attendant Vectoring

Chapter 7: Holiday Vectoring

Chapter 8: Meet-me Conference

Chapter 9: Call Vectoring applications

Appendix A: Call Vectoring commands

Appendix B: Vector management and monitoring

Appendix C: Considerations for the vectoring features

Appendix D: Troubleshooting vectors

Appendix E: Call Vectoring and BCMS interactions

Appendix F: Operation details for the route-to command

Appendix G: Setting up a contact center

Appendix H: Improving performance

About this document

This guide discusses Call Vectoring feature of Avaya Comunication Manager Contact Center.

The following table gives a brief description of each chapter and appendix in this book.

Title/Page	Contents
Call Vectoring overview on page 17	High-level description of vectoring.
Creating and editing call vectors on page 25	Brief tutorial and examples on how to create vectors using the SAT terminal interface.
Call Vectoring fundamentals on page 37	Additional depth of information regarding Call Vectoring and how the feature works.
Call Vectoring applications on page 131	Examples of how Call Vectoring can be implemented on-site.
Basic Call Vectoring on page 55	Detailed information on the Basic Call Vectoring option, including commands specific to the option.
Call Prompting on page 73	Detailed information on the use and implementation of the Call Prompting option, including commands specific to the option.
Attendant Vectoring on page 89	Detailed information on the use of Attendant Vectoring option, including commands specific to the option.
Holiday Vectoring on page 107	Detailed information on the use of the Holiday Vectoring option, including commands specific to the option.
Meet-me Conference on page 117	Detailed information about the commands used to create a Meet-me Conference vector, including an example Meet-me Conference VDN and vector.
Appendix A: Call Vectoring commands on page 153	Complete, detailed list and definition of each vectoring command, including a Job Aid.

Title/Page	Contents
Appendix B: Vector management and monitoring on page 191	Additional information on implementing and upgrading to Call Vectoring.
Appendix C: Considerations for the vectoring features on page 195	Additional considerations for Basic Call Vectoring, Call Prompting, and VDNs.
Appendix D: Troubleshooting vectors on page 199	Error messages.
Appendix E: Call Vectoring and BCMS interactions on page 219	Information regarding how BCMS report on vectoring.
Appendix F: Operation details for the route-to command on page 227	Detailed information on use of the **`route-to`** command.
Appendix G: Setting up a contact center on page 233	Worksheets to assist in the initial set up of a contact center.
Appendix H: Improving performance on page 243	Tips on improving the performance of vectors.

Intended audience and use of the guide

The guide is intended primarily for personnel who use Call Vectoring. You should use this guide as an information source for implementing Call Vectoring. A knowledge of Automatic Call Distribution (ACD) is assumed.

The level of your expertise in Call Vectoring should determine how you use the guide. Users who are unfamiliar with Call Vectoring should read Call Vectoring overview on page 17, then study Creating and editing call vectors on page 25. Finally, advanced users of Call Vectoring may only find it necessary to periodically reference a specific appendix (such as Call Vectoring commands on page 153, which contains a set of Call Vectoring command "manual pages") to get the information needed.

Users who want to set up a contact center should read Appendix G: Setting up a contact center on page 233.

Reasons for reissue

This document is being reissued for the following reasons:

- To be determined.
- Since system capacities change often, the capacity tables have been removed from this document. To determine the maximum values you can use in Call Vectoring commands, see *Avaya MultiVantage Definity Capacities Table*, 555-233-605. To access the document online, go to:

  http://avayadocs.com

Related documents

The following documents may include information related to the ACD feature.

Administration documents

The primary audience for these documents consists of switch administrators who work for external customers and for Avaya's dealers. The satisfaction and needs of our external customers is the primary focus for the documentation.

- *Administrator Guide for Avaya Communication Manager*, 555-233-506 – Provides complete step-by-step procedures for administering the switch, plus feature descriptions and reference information for SAT screens and commands.
- *Avaya MultiVantage Little Instruction Book for Basic Administration*, 555-233-756 – Provides step-by-step procedures for performing basic switch administration tasks. Includes managing phones, managing features, and routing outgoing calls.
- *Avaya Communication Manager Little Instruction Book for Advanced Administration*, 555-233-757 – Provides step-by-step procedures for adding trunks, adding hunt groups, writing vectors and recording announcements.
- *Avaya Communication Manager Little Instruction Book for Basic Diagnostics*, 555-233-758 – Provides step-by-step procedures for baselining your system, solving common problems, reading alarms and errors, using features to troubleshoot your system, and contacting Avaya.
- *Overview for Avaya Communication Manager*, 555-233-767 – Provides a brief description of the features available with DEFINITY ECS. This book does not provide a general overview of the switch nor of basic telephony.
- *Reports for Avaya Communication Manager*, 555-233-505 – Provides detailed descriptions of the measurement, status, security, and recent change history reports available in the system and is intended for administrators who validate traffic reports and evaluate system performance. Includes corrective actions for potential problems.
- *Hardware Guide for Avaya Communication Manager*, 555-233-200 – Provides hardware descriptions, system parameters, lists of hardware required to use features, system configurations, and environmental requirements.

Contact center documents

These documents are issued for Avaya contact center applications:

- *Avaya Communication Manager Contact Center – Guide to ACD Contact Center*, 555-233-516 – Provides feature descriptions and some implementation guidance for contact center features.

- *Avaya MultiVantage Call Center Software Basic Call Management System (BCMS) Operations*, 555-230-706 – Provides information on the use of the BCMS feature for ACD reporting.
- *Avaya MultiVantage Call Center – Call Vectoring Guide for BCS and Guestworks*, 555-233-518 – Provides information on how to write, use, and troubleshoot vectors on Category B products.

How to get help

For those times when you need additional help, the following help services are available. You may need to purchase an extended service agreement to use some of these help services. See your Avaya representative for more information.

- Avaya Centers of Excellence
    - Asia/Pacific
      +65-872-8686
    - Western Europe/Middle East/South Africa
      +441-252-391-889
    - Central/Eastern Europe
      +361-270-5160
    - Central/Latin America/Caribbean
      +1-303-538-4666
    - North America
      1-800-248-1111
- Avaya switch helpline
  1-800-225-7585
- Avaya Toll Fraud Intervention
  1-800-643-2353
- Avaya National Customer Care Center Support Line
  1-800-242-2121

Chapter 1: Call Vectoring overview

Call Vectoring overview provides the following information provides basic terminology and concepts associated with Call Vectoring and summarizes its benefits.

Call Vectoring overview includes the following topics:

What is Call Vectoring?

Call Vectoring is the process of defining vector programs that determine how a specific call should be routed and what call treatment that call is to be given.

Note:

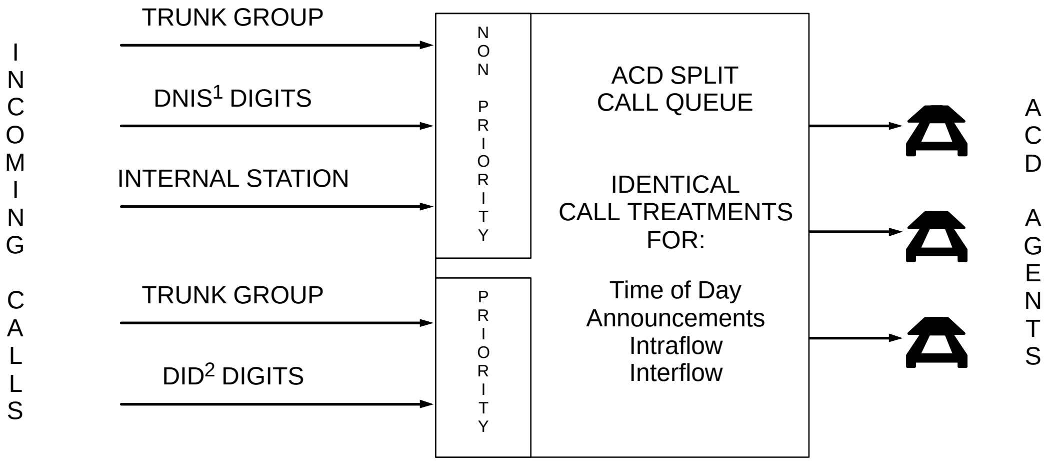
Sample vectors are provided throughout this manual to illustrate vectoring features and capabilities. Because they are simplified to clearly demonstrate specific features, they are not complete and should not be used without modification at your facility.

Call Vectoring provides a highly flexible approach for managing incoming call traffic to the switch. Using vectors, which are a series of user-defined commands, you can direct or route internal and network calls as desired in your contact center and determine how these calls are processed. The processing of calls is known as call treatment. Calls can be directed to on-network or off-network destinations, to ACD agents, or to various other treatments.

Limitations of traditional ACD call processing

The traditional ACD approach is limited in the way it handles queued calls (that is, all calls within a specific queue receive identical announcements, intraflow parameters, and so forth). The following figure shows a simplified illustration of traditional ACD call processing.

Traditional ACD call processing



1. **Dialed Number Identification Service**
2. **Direct Inward Dialing**

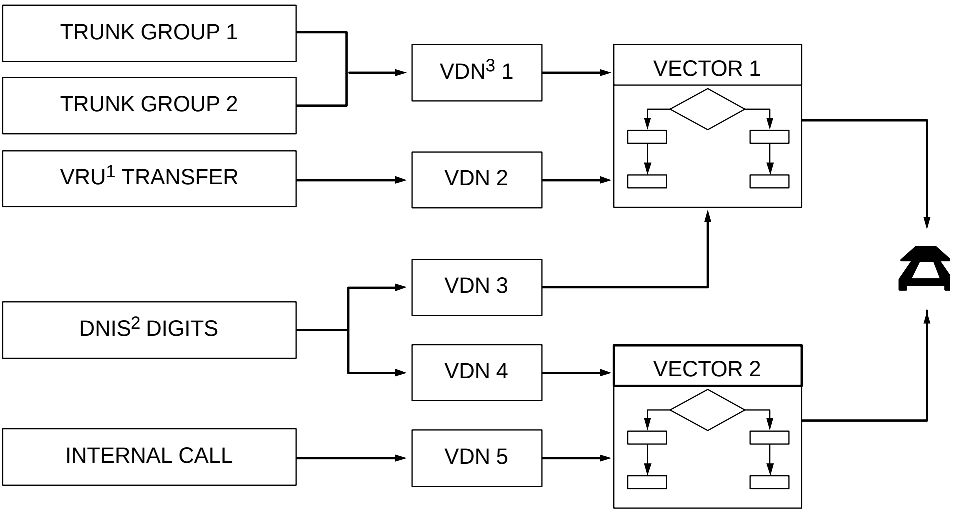
Call Vectoring, on the other hand, permits each call to be treated uniquely according to a number of factors, including the number the caller dials, the number the caller calls from, the number of calls in queue, and the time of day and/or day of the week. This even applies to all calls that are ultimately handled by the same agent group.

Call Vectoring is comprised of three basic components:

- Vector Directory Numbers
- Vectors
- Vector commands

Working together, these components direct incoming calls to the desired answering destinations. They also specify how each call is processed. Call Vectoring may be set up as shown in the following figure.

Use of Call Vectoring for incoming calls



1. **Voice Response Unit**
2. **Dialed Number Identification Service**
3. **Vector Directory Number**

When a call arrives at a switch for which Call Vectoring is enabled, the call is first directed to a Vector Directory Number (VDN). A VDN is an internal telephone number that, in turn, directs the call to a specific vector. The VDN represents the call type or category, for example: billing, customer service, and so on. Thus, it defines the service that is desired by the caller. Multiple VDNs can point to the same or to different vectors, depending on whether the relevant calls are to receive the same or different treatment.

The vector is a set of commands that define the processing of a call. For example, a call can be queued and then routed to another destination.

The following screen shows an example of a vector.

```
1. goto step 3 if calls-queued in split 9 pri l < 20
      2. busy
      3. queue-to split 9 pri l
      4. wait-time 12 seconds hearing ringback
      5. announcement 2921
      6. wait-time 998 seconds hearing music
```

A vector can contain up to 32 command steps. Multiple vectors can be linked together to extend processing capabilities or to process calls to the same or different answering destinations. Any number of calls can use the same multiple vectors and process steps independently.

Understanding your goals and planning your system before you begin writing vectors is crucial. A planning guide is provided in Appendix G: Setting up a contact center on page 233.

Call Vectoring on Category B products

The following call vectoring features are available on Category B products:

- Basic Call Vectoring
- Call Prompting
- Attendant Vectoring
- Holiday Vectoring
- Meet-me Conference

The following Call Vectoring features are not available on Category B products:

- Adjunct Routing
- Advanced Vector Routing
- ASAI Routing (not the same as Adjunct Routing)
- ANI/II Digits Routing
- Best Service Routing
- Call Information Forwarding (CINFO)
- Expert Agent Selection
- G3V4 Enhanced Features
- Look-Ahead Interflow
- VDN of Origin Announcements
- VDN Return Destination.

The following vectoring commands (vector steps) are not available on Category B products:

- adjunct routing
- check skill
- check best
- consider split
- consider skill
- consider best
- converse-on split
- converse-on skill
- messaging skill
- queue-to skill
- queue-to best
- reply-best.

Limited Call Vectoring reports are available by purchasing the separate Basic Call Management System (BCMS) feature.

Call Vectoring options

Call Vectoring provides the following options:

- Basic Call Vectoring provides you with the ability to write vector steps that program the type of processing applied to a call by arranging a set of vector commands in the desired sequence. Depending on the command, you can do the following:
  - Place the call in queue until an agent is available to answer the call.
  - Provide a recorded information or delay announcement to the caller.
  - Allow the caller to leave a recorded message.

  See Basic Call Vectoring on page 55 for more information.
- Call Prompting allows you to collect digits and give some call control to the caller. Specifically, this feature allows callers with touchtone or rotary telephones to enter digits that are subsequently processed by the vector. Among other tasks, Call Prompting allows the caller to do the following:
  - Select one or more options from a menu to access recorded information or be routed to the correct split or agent.
  - Enter an extension to which a call can be routed.
  - Provide the contact center with data, such as a credit card number that the center can use to process the call. This data also can be displayed on the telephone of the agent who answers the call.

  See the Call Prompting on page 73 for more information.
- **Attendant Vectoring** provides you with a means to route calls using call vectoring in an environment other than a traditional contact center.

  See Attendant Vectoring on page 89 for more information.
- **Holiday Vectoring** provides you with a means to apply special call processing on days such as holidays or days when you are having special promotions.

  See Holiday Vectoring on page 107 for more information.
- **Meet-me Conference** provides you with a means to create dial-up, six-party conference calls using the features of Call Vectoring.

  See Meet-me Conference on page 117 for more information.

Benefits of Call Vectoring

Call Vectoring enables calls to be processed at a faster rate within an intelligent, real-time system, thereby providing appreciable cost saving to the user. The following table summarizes the benefits of Call Vectoring.

Call Vectoring benefits

Call Vectoring Benefits	Examples
Call Treatment	
Implement special treatment based on the time of day, the day of the week, and for holidays (for example, routing calls to a different vector when one location is on holiday).	Example application - customer service center on page 132 Conditional branching example on page 69
Automatically change treatment according to either how long the call has been waiting or in response to changing traffic or staffing conditions.	Example application - automated attendant on page 133 Example application - mutual fund company on page 135 Call interflow example on page 67 Using Call Prompting to route by collected digits on page 78 Using Call Prompting to branch by collected digits on page 80
Provide appropriate caller feedback during waiting (for example, music or announcements during heavy calling periods).	Delay announcement example on page 58 Supplementary delay announcement example on page 58 Forced announcement example on page 58 Information announcement example on page 58 Call delay with audible feedback on page 59 Call delay with multiple audio/music source feedback on page 60 Call delay with continuous audible feedback on page 60
Provide multiple and/or recurring informational or delay announcements that are selected according to the time of day/day of the week, call volume, or staffing conditions.	Example application - customer service center on page 132 Leaving recorded messages (VDN as the coverage point option) on page 64 Call interflow example on page 67

Call Vectoring benefits (Continued)

Call Vectoring Benefits	Examples
Provide 24 hour/day, 7 day/week automated information announcements.	Information announcement example on page 58 Call delay with audible feedback on page 59
Remove selected calls (by providing busy or disconnect).	Busy command example on page 62 Call disconnect example on page 62 Leaving recorded message on page 65 Unconditional branching example on page 68
Set up and test, in advance, special call treatments for events such as sales, advertising campaigns, holidays, snow days, and so on.	Information announcement example on page 58 Setting up a Holiday Table on page 111 Holiday Vectoring example 1 on page 114 Holiday Vectoring example 2 on page 114
Provide the caller with a menu of choices.	Example application - mutual fund company on page 135 Using Call Prompting to route by collected digits on page 78 Using dial-ahead digits to bypass announcements, example 1 on page 84
Queue calls to up to three splits simultaneously, consequently improving the average speed of answer and agent productivity.	Example application - customer service center on page 132 Multiple split queuing example on page 63
Implement routing to local or distant destinations.	Example application - customer service center on page 132 Example application - mutual fund company on page 135 Call interflow example on page 67 Using Call Prompting to route by collected digits on page 78 Using Call Prompting to branch by collected digits on page 80
Connect callers to a voice-mail or messaging system either automatically or per caller request.	Example application - mutual fund company on page 135 Leaving recorded messages (VDN as the coverage point option) on page 64 Leaving recorded message on page 65
Call Routing	
Reduce call transfers by accurately routing callers to the desired destination.	Example application - mutual fund company on page 135 Using Call Prompting to route by collected digits on page 78 Using Call Prompting to branch by collected digits on page 80

Call Vectoring benefits (Continued)

Call Vectoring Benefits	Examples
Provide up to four ACD queuing priority levels and the ability to change the queuing priority dynamically, thereby, providing faster service for selected callers.	Example application - customer service center on page 132 Example application - mutual fund company on page 135
Reduce agent and/or attendant staffing requirements by: (1) automating some tasks; (2) reducing caller hold time; (3) having agents in one split service multiple call types.	Example application - mutual fund company on page 135 Information announcement example on page 58 Call delay with audible feedback on page 59 Using Call Prompting to route by collected digits on page 78 Using dial-ahead digits to bypass announcements, example 1 on page 84 Using dial-ahead digits to bypass announcements, example 2 on page 85
Information Collection	
Provide customized and/or personalized call treatment via information collection and messaging.	Example application - automated attendant on page 133 Example application - mutual fund company on page 135 Using Call Prompting to route by collected digits on page 78 Using Call Prompting to select options on page 81 Using dial-ahead digits to bypass announcements, example 1 on page 84

Chapter 2: Creating and editing call vectors

This chapter gives you a practical start writing vectors. In this chapter you will learn the basic information that you need to write a representative vector and enter it online.

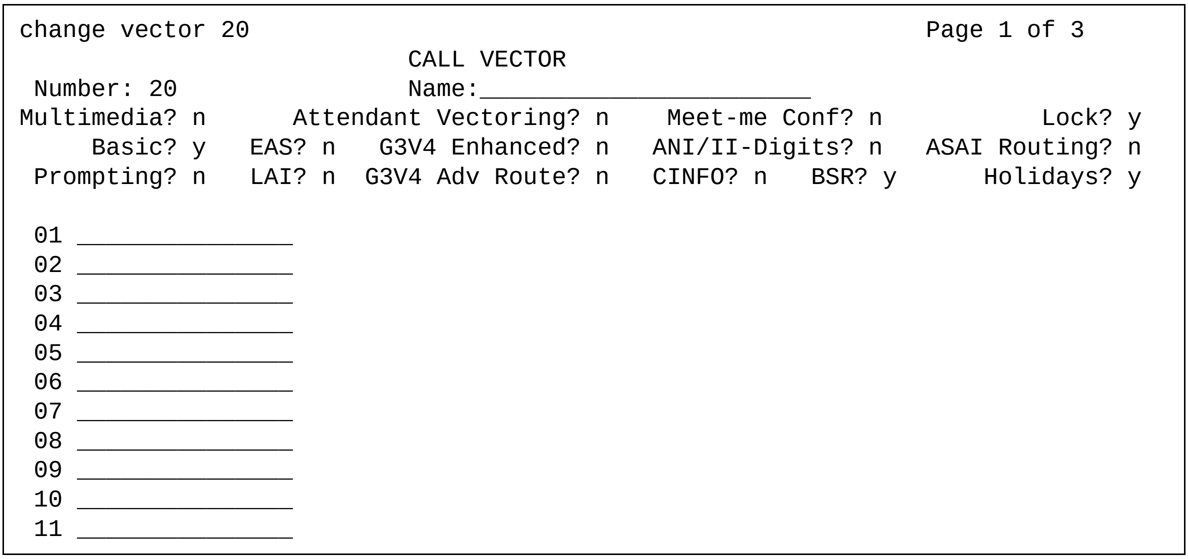
Methods for entering a vector online

- A vector can be entered online using basic screen administration on the system administration terminal by basic screen administration on the system administration terminal.

Call Vector form — basic screen administration

A vector is entered online using basic screen administration by completing the Call Vector form. An example the first page of this form is shown in the following screen example.

Call Vector form (Page 1 of 3)



```
change vector 20                                              Page 1 of 3
                           CALL VECTOR
 Number: 20                Name:_______________________
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? n   LAI? n  G3V4 Adv Route? n   CINFO? n   BSR? y      Holidays? y

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

The following procedure summarizes how you can enter a vector online via basic screen administration.

1. Access the Call Vector Form by executing the change vector ***x*** command, where ***x*** is the number of the vector that you want to access. Use the change vector command either to change an existing vector or to create a new vector.

   If you are not certain of the number or name of a vector, enter the list vector command to view a complete list of all vectors that are administered for your system.

2. Assign a name to the vector by completing the blank next to the `Name` field. The vector name can contain up to 27 alphanumeric characters.

   Note:

   The vector number, which appears next to the `Number:` field, is automatically assigned by the system.

3. In the `Attendant Vectoring` field enter a y if the vector will be used as an attendant vector. Attendant Vectoring can be used only when enabled on the Customer Options form.

4. In the `Meet-me Conf` field enter a y if the vector will be used for the Meet-me Conference feature. Meet-me Conference can be used only when enabled on the Customer Options form.

   Note:

   Both Attendant Vectoring and Meet-me Conference cannot be enabled for a vector at the same time.

5. The Lock field is not used with Category B products.

   Note:

   If Attendant Vectoring or Meet-me Conference is enabled, the Lock field defaults to y and cannot be changed.

6. Look at the next fields and determine where a y (yes) appears. These fields indicate the Call Vectoring features and corresponding commands you can use. If an n (no) appears in one of these fields, you cannot use the corresponding feature.

   Note:

   The Call Vectoring features are optioned from the Customer Options form.

Basic	You can use the Basic Call Vectoring commands. See Basic Call Vectoring on page 55 for details on using these commands.
Prompting	You can use the Call Prompting commands. See Call Prompting on page 73 for details on using these commands.
Holidays	You can create tables to use for special days, such as holidays and promotional days. See Holiday Vectoring on page 107 for information on how to create holiday tables and define holiday vectors.

7. Enter a maximum of 32 vector commands in the blanks next to the step numbers. See Appendix A: Call Vectoring commands on page 153 for a complete description of all Call Vectoring commands.

   Note:

   You need not type every letter of each command that you enter. If you type just the first few letters of a command and press Enter or the Tab key, the system spells out the entire command.

8. Save the vector in the system by pressing Enter.

   Note:

   After editing a vector, verify that the vector will work as intended. This is particularly important if you deleted a step that was the target of a **go-to** step.

Inserting a vector step

To insert a vector step:

1. After entering the `change vector` command, press **F6** (Edit).
2. At the command line, type `i` followed by a space and the number of the step that you want to add and press Enter. For example, to insert a new vector step 3, type `i 3` and press Enter. You cannot add a range of vector steps.
3. Type the new vector step.

When a new vector step is inserted, the system automatically renumbers all succeeding steps and renumbers `goto` step references as necessary. Under certain conditions, attempts to renumber `goto` step references will result in an ambiguous renumbering situation. In this case, the step reference is replaced by an asterisk (*). You will receive a warning indicating that you must resolve the ambiguous references and your cursor automatically moves to the first reference that needs to be resolved. You cannot save a vector with unresolved goto references.

You cannot insert a new vector step if 32 steps are already entered in the vector. However, you can extend the vector program to another vector by using the `goto vector unconditionally` command at step 32.

Deleting a vector step

To delete a vector step:

1. After entering the change vector command, press **F6** (Edit)
2. At the command line, type **d** followed by a space and the number of the step you want to delete and press Enter. You can delete a range of vector steps. For example, to delete steps 2 through 5, type `d 2-5` and press Enter.

When a vector step is deleted, the system automatically renumbers all succeeding steps and renumbers `go-to` step references as necessary. Under certain conditions, attempts to renumber `go-to` step references will result in an ambiguous renumbering situation. In this case, the step reference is replaced by an asterisk (*).

For example, if a vector step that is the target of a goto step is deleted, the goto references are replaced by asterisks (*). For example, if you delete step 7 when you have a `goto step 7 if` vector step, the 7 is replaced by *.

You receive a warning indicating that you must resolve ambiguous references and your cursor automatically moves to the first reference that needs to be resolved. You cannot save a vector with unresolved goto references.

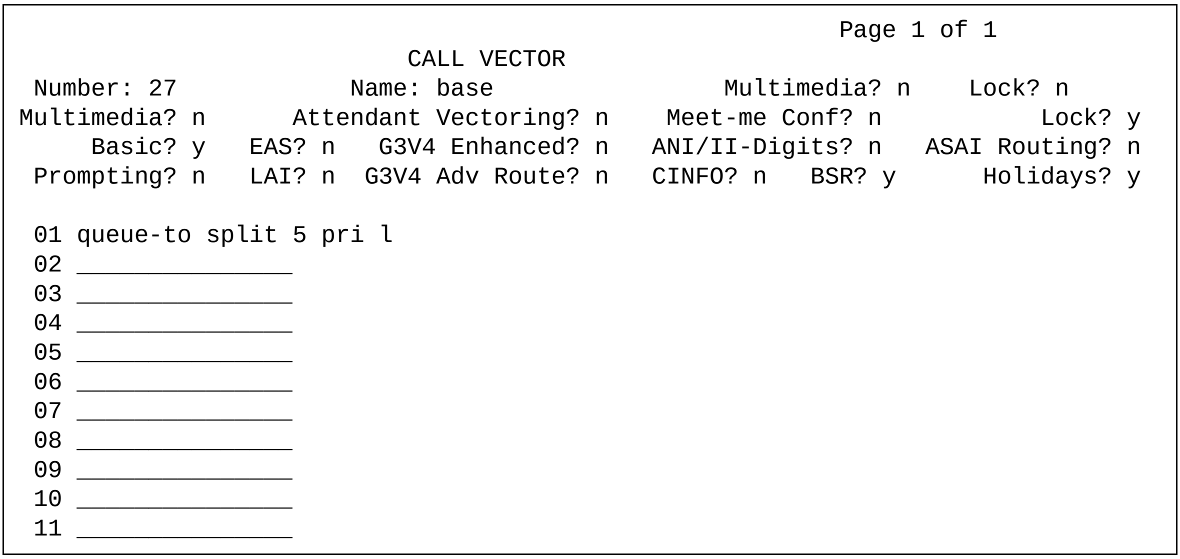
Creating and constructing a vector

Creating and constructing a vector provides a logical approach for vector construction. This method uses a starting vector that consists of one step and then builds on this vector to produce a new vector that provides additional functions. As each step is presented, you are introduced to one or more new vector commands or approaches to vector processing. While it is not practical to present all such commands and approaches, those presented in this tutorial should give you a good idea of how to use Call Vectoring.

Step 1: Queuing a call to the main split

If a call cannot be immediately answered by an agent or operator, the call is usually queued until an agent becomes available. A call can be connected to an available agent or queued via the vector shown in the following example. In this example, calls are queued to Split 5.

Queuing call to main split.



```
                                                                 Page 1 of 1
                                       CALL VECTOR
  Number: 27            Name: base                Multimedia? n    Lock? n
 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? n   LAI? n  G3V4 Adv Route? n   CINFO? n   BSR? y      Holidays? y

  01 queue-to split 5 pri l
  02 _______________
  03 _______________
  04 _______________
  05 _______________
  06 _______________
  07 _______________
  08 _______________
  09 _______________
  10 _______________
  11 _______________
```

Agent Availability

If an agent is available, the **`queue-to split`** command automatically sends the call to the agent without queuing the call. However, if no agent is available, the command queues the call to the main split of agents. Once the call is sent to the main split queue, the call remains there until it is answered by an agent or some other treatment is provided.

Call Priority levels

Each call queued to a split occupies one queue slot in that split. Calls are queued sequentially as they arrive according to the assignment of the priority level. In our vector, note that the priority level low is assigned to the call. The priority level establishes the order of selection for each call that is queued. A call can be assigned one of four priority levels: top, high, medium, or low.

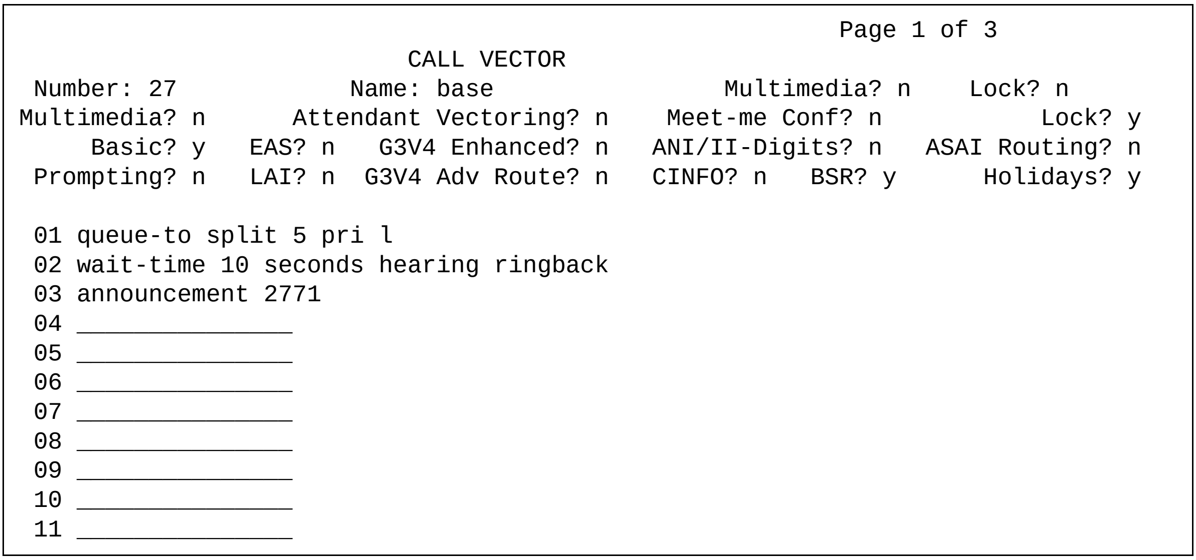
Within a given split (the main split, in our vector), calls are delivered to the agent sequentially as they arrive to the split queue and according to the priority level assigned. Accordingly, calls that are assigned a top priority (if any) are delivered to an agent first, calls that are assigned a high priority are delivered second, and so forth.

Step 2: Providing feedback and delay announcement

A call remains queued until an agent becomes available to answer the call. In the meantime, it is likely that the caller wants to hear some feedback assuring him or her that the call is being processed.

The vector shown in the following example provides one feedback solution. In this example, Announcement 2771 could contain this message: "We’re sorry. All of our operators are busy at the moment. Please hold.”

Providing feedback and delay announcement



```
                                                          Page 1 of 3
                            CALL VECTOR
 Number: 27            Name: base               Multimedia? n    Lock? n
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? n   LAI? n  G3V4 Adv Route? n   CINFO? n   BSR? y      Holidays? y

 01 queue-to split 5 pri l
 02 wait-time 10 seconds hearing ringback
 03 announcement 2771
 04 _______________
 05 _______________
 06 _______________
 07 _______________
 08 _______________
 09 _______________
 10 _______________
 11 _______________
```

Using the wait-time command

The **wait-time** command in step 2 provides a maximum 8-hour delay before the next vector step is processed. The time parameter can be assigned as follows:

- 0-999 secs
- 0-480 mins
- 0-8 hrs

In the example vector, the specified wait time is 10 seconds.

In addition to the delay period, the **wait-time** command provides the caller with feedback. In our vector, **ringback** is provided. Other types of feedback that can be provided with the **wait-time** command are: silence, system music, or an alternate music or other audio source. For more information see, wait-time command on page 59.

The **wait-time** command in the example vector provides the caller with a maximum of 10 seconds of ringback. If an agent answers the call before the **wait-time** command runs its course, the command is terminated, the delay period is ended and the accompanying feedback is stopped. In the current example, if the call is delivered to an agent after 4 seconds the caller does not hear the remaining 6 seconds of ringback.

If the call is not answered by the time the **wait-time** command is completed, vector processing continues.

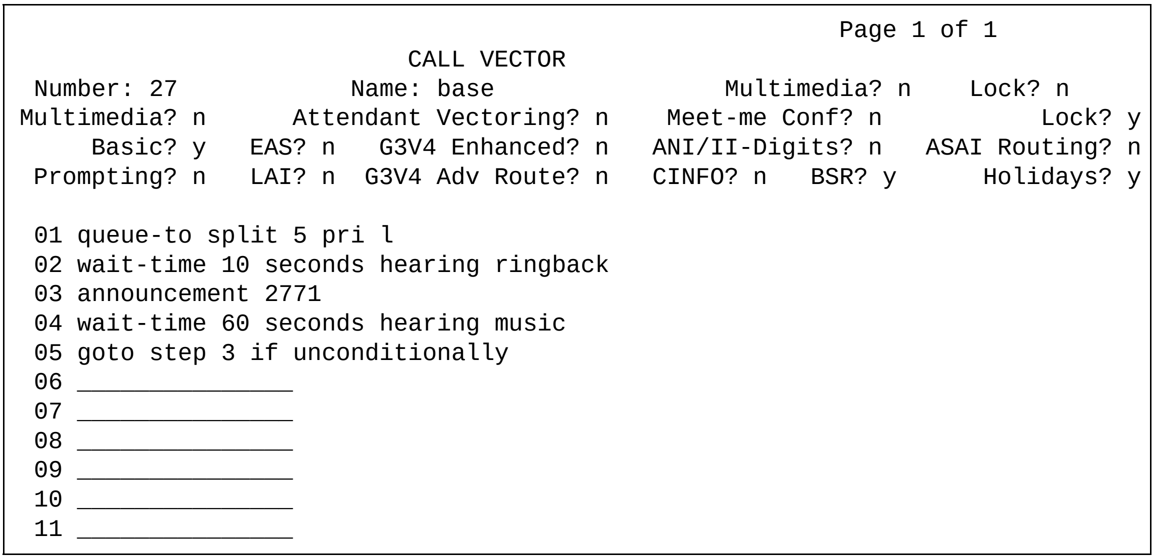
The **announcement** command consists of a recorded message, and it is often used to encourage the caller to stay on the telephone or to provide information to the caller. If a call is delivered to an agent during the **announcement** command, the announcement is interrupted.

Multiple callers can be connected to an announcement at any time. See “Managing Announcements” in *Administrator Guide for Avaya Communication Manager*, 555-233-506, for more information about announcements.

Step 3: Repeating delay announcement and feedback

The announcement vector provides feedback to the caller after the call is queued. However, if the announcement is played and the agent does not answer the call soon after the announcement is complete, further feedback or treatment becomes necessary. One solution is provided in the following Call Vector example.

Repeating delay announcement and feedback



```
                                                                 Page 1 of 1
                              CALL VECTOR
  Number: 27            Name: base                Multimedia? n    Lock? n
 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? n   LAI? n  G3V4 Adv Route? n   CINFO? n   BSR? y      Holidays? y

  01 queue-to split 5 pri l
  02 wait-time 10 seconds hearing ringback
  03 announcement 2771
  04 wait-time 60 seconds hearing music
  05 goto step 3 if unconditionally
  06 _______________
  07 _______________
  08 _______________
  09 _______________
  10 _______________
  11 _______________
```

The **wait-time** command in step 4 of this vector provides additional feedback (music) to the caller. If the call is not answered by the time step 4 is complete, the **goto step** command in step 5 is processed.

Conditional branching

Up to this point, we have discussed and illustrated Call Vectoring commands that cause sequential flow, that is, the passing of vector processing control from the current vector step to the next sequential vector step. The **goto step** command is an example of a Call Vectoring command that causes branching, that is, the passing of vector processing control from the current vector step to either a preceding or succeeding vector step.

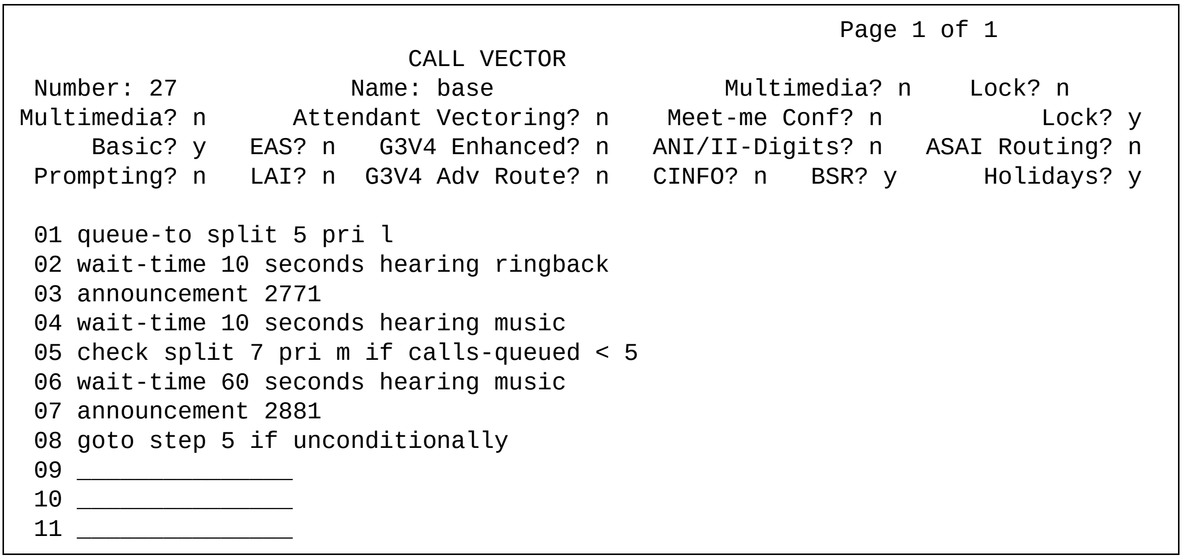
The **goto step** command in vector step 5 allows you to establish an announcement-wait loop that continues until the agent answers the call. Specifically, the command makes an unconditional branch to the **announcement** command in step 3. If the call is not answered by the time that the announcement in step 3 is complete, control is passed to the **wait-time** command in step 4. If the call is still not answered by the time this command is complete, control is passed to step 5, where the unconditional branch is once again made to step 3. As a result of the established loop, the caller is provided with constant feedback.

Step 4: Queuing a call to a backup split

To this point, the vector example involves a call queued to one split. However, Call Vectoring allows a call to be queued to a maximum of three splits simultaneously, which improves can improve overall call response times. Multiple split queuing is especially useful during periods of heavy call traffic.

The vector shown in the following example allows a call to be queued to two splits.

Queuing call to backup split



```
                                                                 Page 1 of 1
                                  CALL VECTOR
 Number: 27            Name: base                Multimedia? n    Lock? n
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? n   LAI? n  G3V4 Adv Route? n   CINFO? n   BSR? y      Holidays? y

 01 queue-to split 5 pri l
 02 wait-time 10 seconds hearing ringback
 03 announcement 2771
 04 wait-time 10 seconds hearing music
 05 check split 7 pri m if calls-queued < 5
 06 wait-time 60 seconds hearing music
 07 announcement 2881
 08 goto step 5 if unconditionally
 09 _______________
 10 _______________
 11 _______________
```

The **queue-to split** command in step 1 queues the call to the main split. But if the call is not answered by the time the **wait-time** command in step 4 is complete, the **check split** command in step 5 attempts to queue the call to backup Split 7 at a medium priority. The condition expressed in the command (**if calls-queued < 5**) determines whether or not the call is to be queued to the backup split. Specifically, if the number of calls currently queued to Split 7 at a medium or higher priority is less than 5, the call is queued to the split.

Conditions used with the check split command

The **calls-queued** condition is one of several conditions that can be included in the **check split** command. The other conditions are **unconditionally**, **available agents**, **staffed agents**, and **oldest call waiting**. As is true for the **queue-to split** command, the **check split** command can queue a call at one of four priorities: **low**, **medium**, **high**, or **top**, except for the oldest-call-waiting condition, which is limited to checking only low priority calls.

Elevating call priority

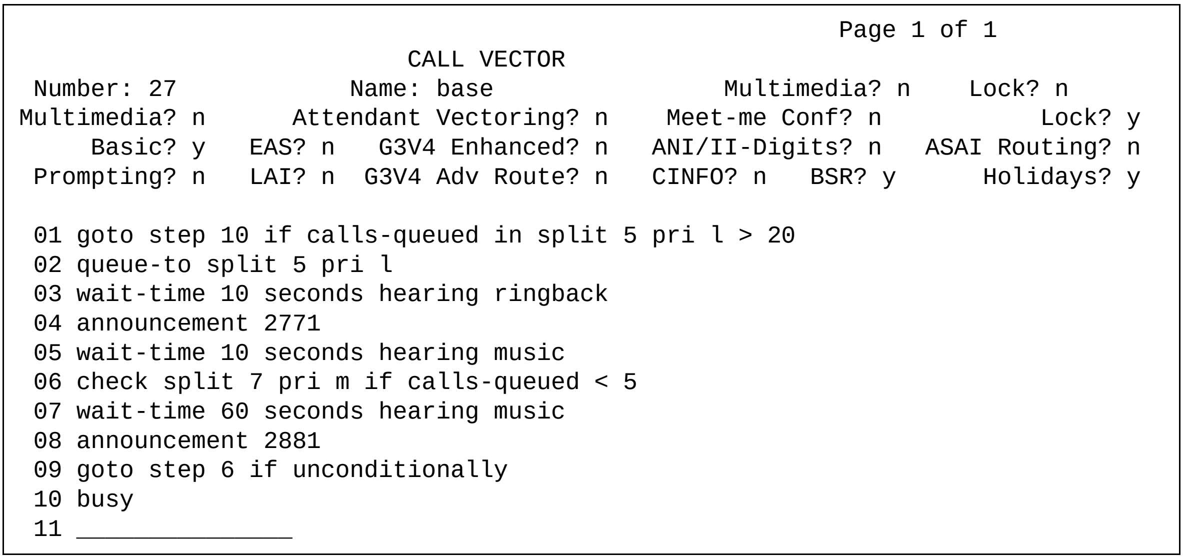
Note that if the call is queued to Split 7, the call priority is elevated from low to medium priority instead of a low priority, which is assigned if the call is queued by the **`queue-to split`** command in step 1. It is a good practice to raise the priority level in subsequent queuing steps to accommodate callers who have been holding the line for a period of time.

Step 5: Checking the queue capacity

A limited number of queue slots can be assigned to each split. Therefore, it is a good practice to check the main split queue for the number of calls that are already queued before allowing another call to queue to the split.

The number of slots assigned to each split is defined in the queue length field on the hunt group screen. A call that attempts to queue to a split with no available queue slots cannot be queued to that split and, accordingly, the **`queue-to split`** command fails. Vector processing would then continue with the next vector step. The following vector example includes provisions for checking queue capacity.

Checking queue capacity



```
                                                           Page 1 of 1
                             CALL VECTOR
 Number: 27            Name: base                Multimedia? n    Lock? n
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? n   LAI? n  G3V4 Adv Route? n   CINFO? n   BSR? y      Holidays? y

 01 goto step 10 if calls-queued in split 5 pri l > 20
 02 queue-to split 5 pri l
 03 wait-time 10 seconds hearing ringback
 04 announcement 2771
 05 wait-time 10 seconds hearing music
 06 check split 7 pri m if calls-queued < 5
 07 wait-time 60 seconds hearing music
 08 announcement 2881
 09 goto step 6 if unconditionally
 10 busy
 11 _______________
```

A check of split 5 is implemented by the **`goto step`** command in step 1. In the example shown above, assume that 21 queue slots are assigned to split 5. Accordingly, the **`goto step`** command tests whether the split contains more than 20 calls via the condition **`if calls-queued in split 5 pri l > 20`**. If this test is successful, control is passed to the **`busy`** command, shown in vector step 10. The **`busy`** command gives the caller a busy signal and eventually causes the call to drop.

Alternately, if 20 or less medium priority calls are already queued to the main split when step 1 executes, the **`queue-to split`** command in step 2 queues the call, and vector processing continues at step 3.

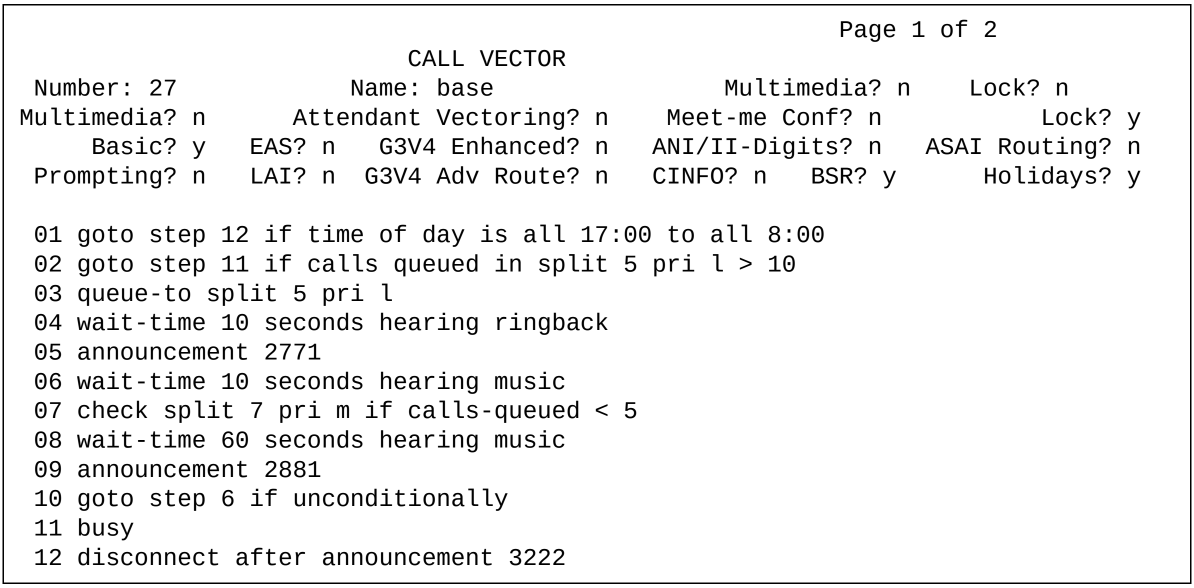
Redirecting calls to a backup split

Instead of providing the caller with a busy tone if the **`queue-to split`** step cannot queue the call, the call can be queued to a backup split. To queue the call to another split, change the step parameter for the **`goto step`** command from 10 to 6 (so that the command reads **`goto step 6.....`**). In this case, control is passed from step 1 to the **`check split`** step (step 6). Because this queuing step is included within a continuous loop of steps (steps 6 through 9), continuous attempts to queue the call are now made.

Step 6: Checking for nonbusiness hours

If a caller calls during nonbusiness hours, you can still provide the caller with some information for calling back during working hours by playing the appropriate recorded message. This strategy is illustrated in the following Call Vector example. This vector would be used for a company that was open 7 days a week, from 8:00 a.m. to 5:00 p.m.

Checking for nonbusiness hours



```
                                                           Page 1 of 2
                            CALL VECTOR
 Number: 27            Name: base               Multimedia? n    Lock? n
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? n   LAI? n  G3V4 Adv Route? n   CINFO? n   BSR? y      Holidays? y

 01 goto step 12 if time of day is all 17:00 to all 8:00
 02 goto step 11 if calls queued in split 5 pri l > 10
 03 queue-to split 5 pri l
 04 wait-time 10 seconds hearing ringback
 05 announcement 2771
 06 wait-time 10 seconds hearing music
 07 check split 7 pri m if calls-queued < 5
 08 wait-time 60 seconds hearing music
 09 announcement 2881
 10 goto step 6 if unconditionally
 11 busy
 12 disconnect after announcement 3222
```

The **`goto step`** command in step 1 checks if the call arrives during nonbusiness hours. Specifically, if the call arrives between 5:00 p.m. and 8:00 a.m. on any day of the week, the command passes control to step 12.

The **`disconnect`** command in step 12 includes and provides an announcement that first gives the caller the appropriate information and then advises him or her to call back at the appropriate time. The command then disconnects the caller.

If the call does not arrive during the specified nonbusiness hours, control is passed to step 2 and vector processing continues. On step 2, split 5 is checked for calls waiting at all priority levels.

Note:

As an alternative to disconnecting callers who place a call during nonbusiness hours, you can allow callers to leave a message by including the **`messaging split`** command within the vector. See Basic Call Vectoring on page 55 for more details.

Chapter 3: Call Vectoring fundamentals

The manner in which a call is processed depends how the switch is implemented and how the Call Vectoring software is implemented on the switch. The success of the call processing relies on:

- The resources that are available to process a call (for example: agents, splits, software, hardware). This is called call management.
- How the call is processed using vector processing, including VDN usage, vector control flow, and intelligent use of the vector programming capabilities.

This chapter describes these fundamental components of Call Vectoring.

Call management

When a call is placed to a switch enabled with Call Vectoring, the call is directed to an appropriate vector by means of a Vector Directory Number (VDN). A VDN is a "soft" extension number that is not assigned to an equipment location. A VDN maps to a single vector, but one or more VDNs can map to the same vector.

Once the call goes to a vector, call routing and treatment are determined by the commands in the vector. Processing starts at the first step and proceeds through the vector. Empty steps are passed over, and the vector process stops after the last step is reached.

However, one vector can direct the call to another vector or VDN, which in turn can direct the call to yet another vector, and so forth, up to a maximum of 1000 vector steps per call. When a call enters vector processing, a loop counter keeps track of the number of vector steps executed. If the loop counter exceeds 1000, a **`stop`** command is executed.

The following sections discuss how calls are routed and queued by way of Call Vectoring. Subsequent sections discuss agent states, priority levels, caller feedback, and caller control.

Call flow

Calls enter a vector and execute steps sequentially beginning with step 1, unless there is a **`goto`** step. Most steps take microseconds to execute. The exception is steps with **`announcement`**, **`wait-time`**, and **`collect digits`** commands. A 0.2-second wait occurs after every seven executed steps unless an explicit wait has occurred. Note that **`wait-time`** with 0 seconds is not an explicit wait.

Call Vectoring uses several call flow methods to redirect and queue calls. These methods involve the use of the Call Vectoring commands, which are described later in this chapter. The methods for queuing and redirecting calls follow:

- Multiple split queuing allows a call to queue to up to three splits.
- Intraflow allows calls that are unanswered at a split within a predefined time to be redirected to one or more other splits on the same switch. If redirection depends on a condition to be tested, the process is referred to as conditional intraflow.
- Interflow allows calls that are directed to a vector to be redirected to an external or nonlocal split destination. This destination is represented by a number that is programmed in the relevant vector. Calls can be routed to an attendant or attendant queue, a local extension, a remote extension (Uniform Dialing Plan (UDP)), an external number, or a VDN.

Note:

This feature is not related to the Look-Ahead Interflow feature. The Look-Ahead Interflow feature is not supported on Category B products.

Caller control

Call Vectoring allows for the temporary transfer of call management control to the caller by several methods:

Caller-Selected Routing – This method prompts the caller to input information in the form of dialed digits from a touchtone telephone or from an internal rotary telephone that is located on the same switch. The capability is available if Call Prompting is enabled. A recorded announcement is usually used for prompting purposes. Once the caller inputs the digits, the call is routed to the correct department or destination. This procedure can significantly reduce the number of transferred calls and thus better satisfy the caller’s needs.

In addition, if Call Prompting and Call Vectoring (CINFO) are enabled, the vector can collect caller-entered digits that are passed from the network by way of an ISDN message. These digits can be used to enhance caller control in the same way as digits that are collected directly by the switch.

Messaging – The caller can leave a voice message in the event that the call cannot be or has not yet been answered. When messaging is enabled, control is eventually passed to the AUDIX split.

Call queuing to splits

Basic Call Vectoring can queue calls to up to three splits simultaneously at any one of four priority levels. This process is called multiple split queuing. The first split to which a call is queued is called the main split, and the second and third split are designated as backup splits. Multiple split queuing enables more efficient utilization of agents, and thus provides better service to callers.

When an agent becomes available in any split to which the call is queued, the following events occur:

- The call is connected to the agent.
- The call is removed from any other queues. Announcements, music, ringback, or other audio source are terminated.
- Vector processing is terminated.

For more information about multiple split queuing, see Multiple split queuing on page 63.

Split queue priority levels

If Call Vectoring is not enabled, queued calls are tracked at one of two priority levels: Medium or High. If a call is queued using Call Vectoring, the call can be assigned one of four priority levels: Top, High, Medium, or Low. Within each priority level, calls are processed sequentially as they arrive.

Note:

If a call is already queued to one or more splits that are currently intended to serve as backup splits, the call could be requeued at the new priority level that is indicated in the command step. For more information on requeuing, see Call Vectoring commands on page 153.

Agent work mode

Call Vectoring can make call management decisions according to real-time agent work modes:

- Staffed-agents considers agents logged in to an ACD split.
- Available-agents considers agents logged in and ready to receive an ACD call.

These work mode states can appear as conditions within the **`check split`** and **`goto`** Call Vectoring commands, so that the commands can be made to check the number of staffed or available agents.

If a hunt group is not monitored, agents in the hunt group do not have log-in, log-out, or work modes. In such cases, staffed-agents is synonymous with administered, and available-agents is the number of agents who are ready to receive a hunt group call.

For ACD calls, agent states are further defined by the relevant work mode. The following list describes these modes:

- After-Call-Work Mode – The agent is unavailable to receive any ACD calls for any split. This mode can be used when the agent is doing ACD call-related work.
- Auto-In Work Mode – The agent is available to receive calls and allows the agent to receive a new ACD call immediately after disconnecting from the previous call. When Multiple Call Handling is enabled, an agent in Auto-In Work Mode can elect to receive ACD calls by placing the active call on hold.

  Note:

  Multiple Call Handling Forced is not supported by Category B products.
- Auxiliary-Work Mode – The agent is unavailable to receive any ACD calls for the specified split. This mode can be used when an agent is performing activities that are not associated with the ACD, such as going on a break.
- Manual-In Work Mode – The agent is available to receive calls. After the agent disconnects from an ACD call, they are automatically puts into the After Call Work Mode.

  Note:

  When Multiple Call Handling is enabled, an agent in Manual-In Work Mode can receive additional ACD calls by placing an active call on hold. For more information about agent work modes and Multiple Call Handling, see *Avaya Communication Manager Contact Center – Guide to ACD Contact Center*, 555-233-516.

Calling party feedback

The initial feedback a caller hears as the call is being processed by a vector depends on the origin classification of the call, which can be one of the following:

- Internal call from another switch user.
- Non-CO incoming call over a DID or tie trunk over which incoming digits are received.
- CO incoming call over a CO or automatic type tie trunk over which no digits are received.

For an internal or a non-CO call, the caller hears silence until one of the following vector steps is reached:

- For **`wait`** commands with system music, ringback, or an alternate music or audio source, the caller hears system music, ringing, or the music or audio associated with an administered port.
- For any **`announcement`** command, the caller hears the specified announcement. command is processed.

- For a **`busy`** command, the caller hears a busy signal.
- When the call rings a station, the caller hears ringback.

For a CO call, the caller hears CO ringback until one of the following vector steps is reached:

- Announcement (Caller hears the announcement.)
- Wait with system music or alternate audio/music source (Caller hears system music, or the music or audio associated with an administered port.)
- Call answered (Caller hears the agent or voice response answering the call.).

For a CO call that has answer supervision already supplied by way of the processing of an **`announcement`** or the issuing of a **`wait-time`** command, the caller may hear any of the following:

- Announcement when any **`announcement`** command is processed.
- Ringback, silence, system music, or an alternate audio or music source when a **`wait-time`** command is processed.
- Busy when a **`busy`** command is processed.
- Ringback when the call rings at a station.

Examples of how subsequent caller feedback is provided in a vector are provided in Basic Call Vectoring on page 55.

Dialed number identification service (DNIS)

In the traditional ACD arrangement, each agent in a given split is trained to answer calls that are relevant to one specific purpose. However, a contact center may wish to utilize agents trained to address multiple types of calls. This arrangement can allow resources to be used in a more efficient manner, with fewer agents overall and less administrative intervention by the ACD manager. For example, where 5 agents might be needed in each of three smaller splits (15 agents total) to handle 3 types of calls, only 11 or 12 agents might be needed in the combined split.

A network service known as Dialed Number Identification Service (DNIS) is available to exploit multi-skill agent capabilities. DNIS enables a unique multidigit number based on the dialed number associated with the call. The unique number may be sent to an agent, used to provide different treatments for the call, and so forth.

The DNIS number is a function of the telephone number dialed by the caller. Each DNIS number in your telephone system can be programmed to route to an ACD split that is comprised of agents who are proficient in handling several types of calls.

Call Vectoring takes the DNIS number from the network and interprets this number as a VDN. When the call is delivered to the agent terminal, the unique name that is assigned to the particular VDN is displayed on the agent's terminal. This allows the agent to know the specific purpose of the call. As a result, the agent can answer with the appropriate greeting and be immediately prepared to service the customer.

Vector processing

If Call Vectoring is in effect, telephone calls are processed by one or more programmed sequences of commands called vectors.

Vector processing includes the following topics:

- Vector Directory Number (VDN)
- Vector control flow
- Programming capabilities.

Vector Directory Number

Within Call Vectoring, calls access the appropriate vector(s) by way of a Vector Directory Number (VDN). A VDN is a “soft” extension number that is not assigned to an equipment location. In effect, the digits dialed by a caller or sent to the switch from an external network are translated within the system as a VDN.

The VDN points to the vector, and it defines the service desired by the caller. The VDN also serves as the application number. It allows for specific call-handling and agent-handling statistical reporting within the Basic Call Management System (BCMS).

VDNs are assigned to different vectors for different services or applications that require specific treatments. Any number of VDNs can point to the same vector. As a result, the same sequence of treatments can be given to calls that reach the system from different numbers or from different locations.

The VDN has several properties. These properties are administered on the Vector Directory Number form. The following screens show all possible fields available for a VDN. However, some fields display only when certain features are enabled.

```
change vdn xxxxx                                                  page 1 of 2
                      VECTOR DIRECTORY NUMBER

                            Extension: 2001
                                 Name: vdn 2001
                        Vector Number: 1
                  Attendant Vectoring? n
                   Meet-me Conference? n
                   Allow VDN Override? n
                                  COR: 1
                                   TN: 1
                             Measured: internal
       Acceptable Service Level (sec): 20
        VDN of Origin Annc. Extension:
                            1st Skill:
                            2nd Skill:
                            3rd Skill:
```

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

                                  Audix Name:
                       Messaging Server Name:
                          Return Destination:

                      VDN Timed ACW Interval:
                             BSR Application:
                BSR Available Agent Strategy: 1st-found
                      Conference Access Code:
                       Conference Controller:
                Display VDN for Route-To DAC?
   VDN Override for ISDN Trunk ASAI Messages?
```

The following list describes every field that is available on the VDN form and gives you information on the values that can be entered in the field.

- **Extension** — The extension number used to identify the VDN.
- **Name** — An alphanumeric name that identifies the VDN. This is an optional field that need not contain any data. The name may be truncated on agents' displays depending on the application. The default is blank.
- **Vector Number** — An identification number that determines which vector is activated when a call comes into a VDN. Several VDNs may send calls to the same vector. The maximum Vector Number capacity varies by switch. For more information, see the Avaya switch capacity documents, which can be accessed online at: http://www.avayadocs.com

- **Attendant Vectoring** — A y indicates that this is an Attendant Vectoring VDN. For more information, see Attendant Vectoring on page 89. This field defaults to y if Attendant Vectoring is the only Call Vectoring feature enabled on the customer options form. When removing a VDN, the switch verifies that this VDN is not being used on either the Console Parameters form or the Tenant Partitioning forms.
- **Meet-me Conference** — A **y** indicates that this is a Meet-me Conference VDN. For more information, see Meet-me Conference on page 117.
- **Allow VDN Override** — Valid entries are **y** and **n** (default). If a call is processed through multiple VDNs, this entry may affect the VDN name and various options assigned to the VDN. If it is set to n, the name of this VDN appears on the agent's display and the VDN's AUDIX mail is accessed. If any subsequent VDNs are used to process this call, their names do not appear on the terminating display and the AUDIX mail for the original VDN is accessed. If the field is set to y, the name of the VDN that appears on the terminating display depends on the administration and chaining of the subsequent VDNs and the AUDIX mail for the last VDN that is accessed. The default is n.
- **COR** — A 1-digit to 2-digit number that specifies the class of restriction (COR) to be assigned the VDN. The default value is 1. The field cannot be blank and must have an entry in the range from 0 to 95.
- **TN** — Enter the Tenant Partition number. The default value is 1.
- **Measured** — Used to collect measurement data for this VDN.Valid entries for Category B are **`internal`** or **none**. Internal data collection is done by BCMS. The default is none.

  Note:

  The BCMS feature must be enabled on the System-Parameters Customer-Options form for the Measured field to be set to `internal`.
- **Conference Access Code** — A 6-digit Meet-me Conference access code or blank. Once a conference access code is assigned, an asterisk displays in this field for subsequent change display or remove operations by all users except the "init" superuser login. This field is displayed only when the VDN is a Meet-me Conference VDN.
- **Conference Controller** — A valid extension number or blank. 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. This field is displayed only when the VDN is a Meet-me Conference VDN.

VDN Override for ISDN Trunk ASAI Messages – This feature allows users to associate Called Number ASAI message information with the active VDN, instead of the actual Called Number information that is included in the SETUP message of an incoming ISDN call. This functionality is useful in the following situations:

- Incoming ISDN/PRI calls are routed to a VDN that prompts callers to enter digits from a list of options typically presented to the caller via a vector-initiated recorded announcement
- The entered digits routes the call to a new VDN

For this or other types of vector call-flows where the call is routed to another VDN and the "VDN Override" feature is active for the call, the "active VDN" extension is sent as the “Called Number” to CTI applications for the following message types:

- "Call Offered" ASAI event message
- “Alerting" ASAI event message
- "Queued" ASAI event message
- “Connect" ASAI event message
- "Adjunct Route-Request" ASAI message

This option is displayed only if the following conditions are met:

- The G3 Version specified on the System-Parameters Customer Options form is set to v10 or later.
- The ASAI Link Core Capabilities option in the System-Parameters Custom-Options screen is set to **y**.

This option can be set to y only if the Allow VDN Override? on page 1 of the Vector Directory Number is also set to **y.**

Interactions with the VDN Override for ISDN Trunk ASAI Messages? feature include the following:

- The initial active VDN extension associated with a call is overridden by the extension of a secondary VDN if the call is subsequently routed by a route-to number vector step.
- The active VDN information is not preserved if the call is answered by an ACD agent or station user, and subsequently transferred or conferenced to another station or outgoing trunk user.

Implementation notes

VDNs can be preassigned to incoming trunk groups, or they can be sent in digit form to the switch by a public or private network. The digits that are sent to the switch can come from the serving Central Office (CO) or toll office by way of the Direct Inward Dialing (DID) feature or DNIS. The digits can also come from another location by way of dial-repeating tie trunks, or they can be dialed by an internal caller. For a non-ISDN call, the last four digits of the number are sent to the system. For an ISDN call, the entire 10-digit number is sent to the system.

The last few digits of the destination passed to the switch/ACD on a DID or DNIS or on a dial tie-trunk call comprise the VDN. Automatic trunks do not pass destination address digits. Instead, each such trunk always routes to a specific incoming destination that is programmed for the corresponding automatic trunk group. The destination can be an attendant queue, an extension, a hunt group number, or a VDN.

VDN Override

VDN Override allows information about a VDN to which a call is routed to be used instead of the information about the current VDN. This information includes:

- The name of the subsequent VDN
- Messaging split command with the “active” entry.

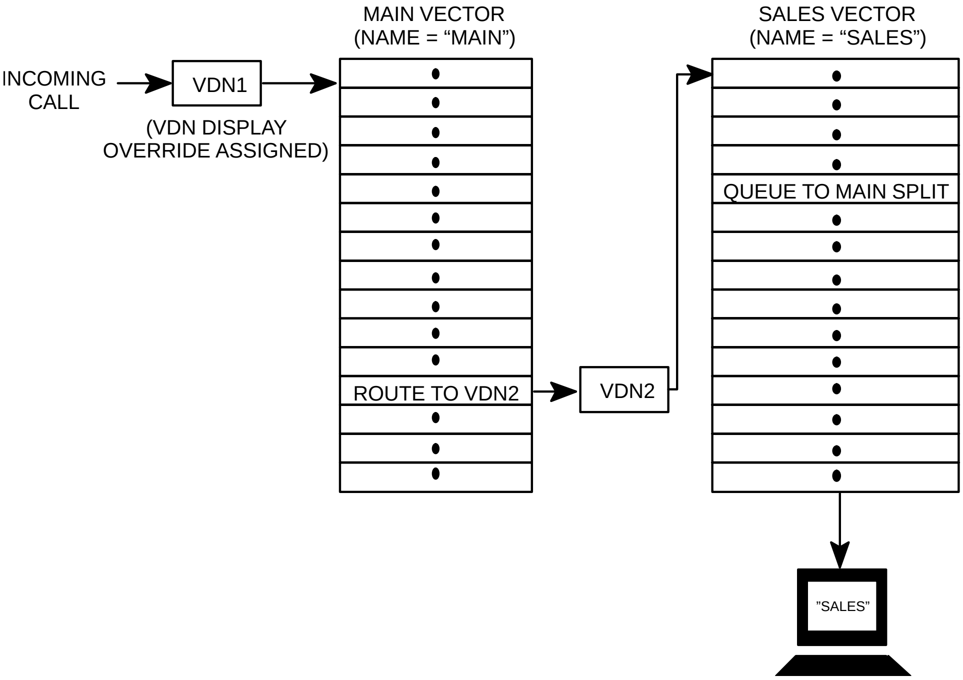
Note:

> Throughout this document the “active” VDN is the active called VDN as modified by VDN override rules. The “latest” VDN is the most recent VDN to which the call was routed.

VDN Override can be used in conjunction with a vector that prompts the caller for a particular service. For example, a call is placed to an automobile dealer. Like most such dealers, this one consists of several departments, including “Sales” and “Parts.” Assume that the caller wants to talk to someone in “Sales.” In this case, the call comes into the “Main” vector (whose VDN name is “Main”) and is eventually routed to the “Sales” vector (whose VDN name is “Sales”). If VDN Override is assigned to the “Main” VDN, the “Sales” VDN name appears on the agent’s telephone display when the call is finally connected to the agent.

The VDN Override process is shown in VDN override assigned to originally called VDN on page 47. In this example, the “Sales” VDN is the active VDN as well as the latest VDN. If VDN override was not assigned to the “Main” VDN, the agent’s telephone display would show “Main.” In this case, “Main” is the active VDN while “Sales” is the latest VDN.

VDN override assigned to originally called VDN



VDN in a coverage path

A VDN can be assigned as the last point in a coverage path. Whenever a VDN is assigned as such, a call goes to coverage and can then be processed by Call Vectoring or Call Prompting if either is enabled. Accordingly, the Call Coverage treatment for the call is extended. Coverage can be sent to an external location or the type of coverage can be controlled by the caller.

VDN in a coverage path is used for a number of applications, including:

- Routing coverage calls off-premises using the **`route-to`** command.
- Serving as a coverage point for specific call operations. For example, sending calls to a secretary during the day and to AUDIX at night.

For more information, see Option with the VDN as the coverage point on page 64. For information about interactions, see *Administrator Guide for Avaya Communication Manager*, 555-233-506.

Redirect on No Answer to a VDN

The Redirection on No Answer (RONA) feature redirects a ringing ACD call after an administered number of rings. It prevents a call from ringing indefinitely at a terminal when an agent does not answer. When a call is redirected, the system puts the agent into AUX work so that the agent is no longer available to receive ACD calls. In the case of Auto-Available Splits, the system logs the agent out when a call is redirected.

A VDN can be administered as the destination of a RONA processed call. A call that is not answered can be redirected to a VDN to receive special treatment. Enter the number of the destination VDN for a RONA call in the Redirect to VDN field on the Hunt Group form. All calls that are redirected by RONA from that split are sent to the same administered VDN.

If no destination VDN is administered, but the number of rings for redirection is entered, the call redirects back to the split.

For more information, see the Redirection on No Answer section in *Avaya Communication Manager Contact Center – Guide to ACD Contact Center*, 555-233-516.

Vector control flow

The vector process starts at the first step in the vector and then proceeds sequentially through the vector unless a **`goto`** command is encountered. Any steps that are left blank are skipped, and the process automatically stops after the last step in the vector.

The Call Vectoring "programming language" provides three types of "control flow that pass vector-processing control from one vector step to another.The types of control flow are described in the following list:

- Sequential flow passes vector-processing control from the current vector step to the following step. Most vector commands allow for a sequential flow through the vector.

  Note:

  Any vector command that fails automatically passes control to the following step.

- Unconditional branching unconditionally passes control from the current vector step to either a preceding or succeeding vector step or to another vector. For example, **`goto step 6 if unconditionally`**.
- Conditional branching conditionally passes control from the current vector step to either a preceding and/or succeeding vector step or to a different vector. This type of branching is based on the testing of threshold conditions. For example, **`goto vector 29 if staffed-agents in split 6 < 1`**.

  Note:

  Call Vectoring has an execution limit of 1000 steps. Once a call enters vector processing, a "loop counter" keeps track of the number of vector steps executed. If the loop counter exceeds 1000, a **`stop`** command is executed.

  Note:

  An implicit wait of 0.2 seconds is provided after every seven vector steps if vector processing is not suspended during any one of these steps. For more information, see Wait-time command on page 186.

Termination versus stopping

When vector processing is terminated, the call leaves the vector. Vector termination can result from a number of events, such as when a call is:

- Ringing at an agent's station
- Abandoned by the calling party
- Subject to a forced disconnect or a forced busy
- Successfully routed to an extension or to an off-premises number

The termination of vector processing termination differs from stopping, which is caused by the **`stop`** command or by the execution of the final step in the vector. Termination differs from stopping in the following ways:

- If a call is queued, termination removes the call from the queue.
- A **`stop`** command prevents the processing of new vector steps but leaves the call in queue and the calling party continues to receive feedback, such as ringback.
- If vector processing stops and the call is not queued, the call is dropped.

Programming capabilities

Call Vectoring commands perform various call-related functions, which include:

Providing call treatments

Audible feedback, including silence, ringback, system music, or an alternate audio or music source, or a busy tone can be provided to the caller. The caller can be provided with a recorded announcement to indicate that an agent is unavailable to answer the call or to provide other information or instructions. An Audix session can also be initiated.

Vector processing can be delayed for a specific number of seconds before the next vector step is executed. The call can also be disconnected, if necessary.

- Routing calls. Calls that are not immediately answered by an agent can be queued to one or more splits. A caller can also leave a recorded message if he or she chooses to do so. Finally, a call can be routed to a number programmed in the vector or to digits that are collected from the caller.
- Branching/programming. Branches can be made from one vector step to another such step or to another vector. This can be done unconditionally as well as conditionally. Conditional branching is done according to a number of conditions, for example, number of available agents in a split, number of calls in a split queue, the number of the phone the call is made from, and so forth. Finally, vector processing can be stopped when necessary.

- Collecting and acting on information. Optionally, touchtone digits can be collected and serve as the basis for further vector processing. For example, the caller can enter certain touchtone digits to reach a specific agent.

Command summary

This section lists and describes the commands used by the Call Vectoring features. The list is meant to help familiarize the reader with these commands. The commands are also described in further detail in Call Vectoring commands on page 153.

- Announcement provides the caller with a recorded announcement.
- Busy gives the caller a busy signal and causes termination of vector processing.
- Check conditionally checks the status of a split for possible termination of the call to that resource. The command either connects to an agent in the split or puts the call into its queue at the specified queuing priority level if the condition specified as part of the command is met. A call can be queued to up to three different splits simultaneously.
- Collect Digits collects up to 16 digits that are either entered by the caller during vector processing or sent by the network. An optional announcement can be played first when the digits are being collected directly from the caller.
- Disconnect ends treatment of a call and removes the call from the switch. The command also allows the optional assignment of an announcement that will play immediately before the disconnect.
- Goto Step is a branching step that allows conditional or unconditional movement to a preceding or succeeding step in the vector. Conditional branching is determined by a number of factors. For example: the number of calls that are queued in the split, the number of staffed agents who are in the split, if the call arrives at a time of day that is in a holiday table, and so on.
- Goto Vector is a branching step that allows conditional or unconditional movement to another vector. Conditional branching is determined by a number of factors. For example: the number of calls that are queued in the split, the number of staffed agents who are in the split, if the call arrives at a time of day that is in a holiday table, and so on.
- Messaging Split allows the caller to leave a message for a specified extension or the VDN extension.
- Queue-to unconditionally queues a call to a split and assigns a queuing priority level to the call in case no agents are available. A call that is sent with this command either connects to an agent in the split or enters its queue.
- Queue-to attd-group queues a call to a specified attendant group and is available only for attendant vectors. A call that is sent with this command either connects to an available agent within the group or enters the queue if no agent is available.
- Queue-to attendant queues a call to a specific attendant and is available only for attendant vectors. The call only queues to the agent if the agent is a member of the TN associated with the call.

- Queue-to-hunt group queues a call to up to three hunt groups. A call that is sent with this command connects to an agent in the hunt group or enters the hunt group queue.
- Route-to Digits routes the call to the destination that is specified by a set of digits that are collected from the caller by the previous **`collect digits`** step. For more information, see Appendix F: Operation details for the route-to command on page 227.
- Route-to Number routes the call to the destination specified by the administered digit string. Form more information, see Appendix F: Operation details for the route-to command on page 227.
- Stop terminates the processing of any subsequent vector steps.
- Wait-Time is used to specify whether the caller hears ringback, system music, silence, or an alternate audio or music source while the call is waiting in queue. The command also delays the processing of the next vector step by the specified delay time that is included in the command's syntax.

Condition testing within the commands

As was mentioned in the previous section, a number of the Call Vectoring commands are implemented according to a tested condition that comprises part of the command. In other words. If the condition that is expressed in the command is true, then the command action is executed. If the condition that is expressed in the command is false, then the command action is not implemented, and the next vector step is processed.

For more information about the syntax of each condition, see Appendix A: Call Vectoring commands on page 153.

The following list provides a set of conditions that might comprise the conditional portion of a Call Vectoring command:

- The number of staffed agents in a split
- The number of available agents in a split
- The number of calls queued at a given priority to a split
- The amount of time that the oldest call has been waiting in a split
- Whether or not a call receives special holiday processing
- The number of active calls that have been routed by a VDN
- The digits entered by the caller
- The time-of-day and day of the week that the call is placed. The syntax for this condition can be illustrated as follows: **`mon 8:01 to fri 17:00`** means anytime between 8:01 a.m. Monday through 5:00 p.m. Friday, and **`all 17:00 to all 8:00`** means between 5:00 p.m. and 8:00 a.m. on any day of the week.

Depending upon the condition, you can use the following comparison operators with Category B products:

- For comparing digits collected, you can only use **=** (equal to).
- For comparing to agent status (for example: available agents, login agents, calls in queue), you can use **<** (less than), **>** (greater than), and **=** (equal to).
- For comparing whether or not a call receives special holiday processing, you can use “in” or “not-in.”

Note:

Category B products do NOT support the enhanced set of comparators, such as >=, <=, and <>.

The chapters on the Call Vectoring features illustrate condition checking in more detail.

Using VDN COR Restrictions to limit outside access

Routing calls through the switch with Call Vectoring can raise some security issues. For more information on security issues, refer to the *BCS Products Security Handbook*.

A VDN has a Class Of Restriction (COR). Calls processed by the vector carry the permissions and restrictions associated with the COR of the VDN.

For example, if a vector in the switch 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 Feature Access Codes, or most other sets of dialed digits. To deny incoming callers access to outgoing facility paths, the COR of the Vector Directory Number must be configured to disallow outgoing access. This should 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, and blocking access to specific CORs assigned to outgoing Trunk Groups using the Calling Permissions section of the Class of Restriction Screen.

Review the Classes of Restriction assigned to your VDNs. If they are not restricted, consider assigning restrictions on the VDN and/or using **goto** tests on those digits to prevent callers from exiting the system via the vector.

Chapter 4: Basic Call Vectoring

Basic Call Vectoring allows you to use various vector commands to program specific call treatments. The vector commands that are available to you as part of the Basic Call Vectoring feature set are the simplest and most common commands that are used to program call vectors.

Vector commands can direct calls to various destinations, such as splits or other vectors. The commands can also direct calls to various treatments, such as announcements, a forced disconnect, a forced busy, or a delay treatment.

Basic Call Vectoring includes the following topics:

Command set

The following table shows the commands used for Basic Call Vectoring.

Command category	Action taken	Command
Treatment		
	Play an announcement.	**`announcement`**
	Delay with audible feedback of silence, ringback, system music, or alternate audio or music source.	**`wait-time`**
	Play a busy tone and stop vector processing.	**`busy`**
	Disconnect the call.	**`disconnect`**
Routing		
	Queue the call to an ACD split.	**`queue-to split`**
	Queue the call to a backup ACD split.	**`check split`**
	Leave a message.	**`messaging split`**
	Route the call to a number that is programmed in the vector.	**`route-to number`**
Branching/Programming		
	Go to a vector step.	**`goto step`**
	Go to another vector.	**`goto vector`**
	Stop vector processing.	**`stop`**

Basic Call Vectoring allows you to use vectoring commands from each of the above Command Categories to process telephone calls. The following sections explain the commands in more detail.

Treatment commands

Call treatment is the type of feedback the caller receives if the caller is not immediately connected to an agent. Basic Call Vectoring includes the ability to implement several types of call treatment commands.

announcement command

The **`announcement`** command connects calls to a recorded announcement.

Announcements can be classified into three groups:

- Delay announcements
- Forced announcements
- Information announcements

Depending on the type of announcement equipment and how the equipment is administered, callers may be required to listen to an entire announcement or they may be able to interrupt an announcement as it is playing.

When a call is connected to an announcement, any previous treatment is discontinued.

For announcements that always start at the beginning, the caller may have to wait in an announcement queue if the announcement is not ready to play. Callers hear the previously established call treatment (if any) until the announcement starts. If the announcement queue is full, vector processing retries the **`announcement`** command indefinitely.

Note:

> If an integrated announcement board is in use and the requested announcement is not administered or recorded, vector processing skips the **`announcement`** command and continues with the next vector command.

If the call is in a split/skill queue, the call remains in queue while the announcement plays. If the call is still in queue after the announcement ends, the caller hears silence until another **`announcement`** command, a **`wait hearing ringback`** command, or a **`wait hearing music`** command is processed. If the call connects to a station while the announcement is playing, the announcement stops and the caller hears ringback.

When the announcement completes and is disconnected, the caller hears silence until either a vector step with alternate treatment is processed or the call reaches an agent's station.

Delay announcements

An example of a delay announcement is shown in the following figure.

Delay announcement example

```
announcement 2556 (‘‘All our agents are busy.
     Please hold.’’)
```

If the caller remains on hold, a supplementary delay announcement similar to the following example can be used.

Supplementary delay announcement example

```
announcement 2557 (‘‘Thanks for holding.  All
     our agents are still busy.  Please hold.’’)
```

A delay announcement is usually coupled with a delay step, which is provided by the **`wait-time`** command. For more information about the **`wait-time`** command, see wait-time command on page 59.

Forced announcements

In some circumstances, a contact center may not wish to allocate resources to certain calls. Usually, this option is exercised when heavy call traffic is expected due to a major event such as a widespread service problem which is currently being addressed. An appropriate announcement can be inserted into a vector to address such circumstances. An example of such a forced announcement is shown below.

Forced announcement example

```
announcement 1050 (‘‘We are aware of the current
     situation and are working to rectify the problem.  If your
     call is not urgent, please call back later.’’)
```

Information announcements

In some circumstances, a contact center can anticipate that the caller can be provided with recorded information that fully addresses their needs, so that no further interaction is required. An example of such an announcement, which is referred to as an information announcement is shown in the following example.

Information announcement example

```
disconnect after announcement 2918 (‘‘Today has
     been declared a snow day.  Please report for work tomorrow
     at 8 A.M.’’)
```

A forced announcement is followed by a **disconnect** command is used with the announcement. After the announcement, the caller is disconnected, since he or she need not stay on the line any longer.

wait-time command

The **wait-time** command enables you to create a vector that delays the call with audible feedback. In presenting an example of a delay announcement earlier in this chapter, we mentioned that this type of announcement is usually coupled with a delay step. A delay step is provided by the **wait-time** command, which allows the caller to remain on hold for at least the amount of time that is indicated in the command.

The following example shows an announcement that includes the **wait-time** command in a delay step.

Call delay with audible feedback

```
announcement 2556 (‘‘All of our agents are busy.
     Please hold.’’)
     wait-time 20 seconds hearing music
```

In the example shown above, the caller waits at least 20 seconds for the call to be answered by an agent. During this wait period, the caller is provided with system music, which is one type of feedback that is available with the **wait-time** command.

If the delay step is the final effective step in the vector, the audible feedback continues beyond the specified duration. In a vector, a final effective step is defined as the last vector step, or a vector step that is followed by a **stop** step.

Under normal circumstances, the audible feedback continues until the call is either answered or abandoned. However, if the call is not queued when vector processing stops, the call is dropped. Feedback also continues during the wait period before the connection of an announcement and/or a Touch-Tone Receiver (TTR).

For more information about TTRs, which are used with the Call Prompting feature, see Chapter 5: Call Prompting on page 73.

Multiple audio or music sources on delay

You can specify an alternative audio or music source for a vector **wait-time** step. This alternative source can be any extension number that is administered on the Announcements/Audio Sources form. For instructions for entering an audio or music source on this form, see *Administrator Guide for Avaya Communication Manager*, 555-233-506.

With the Multiple Audio/Music Sources feature, you can tailor the **wait-time** feedback to the interests, tastes, or requirements of the audience. You can provide specific types of music or music with overlays of advertising that relate to the service provided by the splits or skills that the vector serves. Or, additional advertising messages can be heard by the callers as they wait for an available agent.

An example of an announcement that includes an alternative audio or music source in the **`wait-time`** step is shown below.

Call delay with multiple audio/music source feedback

```
announcement 2556 (‘‘All of our agents are busy.
     Please hold.’’)
     wait-time 20 seconds hearing 55558 then music
```

When the **`wait-time`** step is encountered, the caller is connected to extension 55558 for 20 seconds. At the end of 20 seconds, the next vector step is executed. The “then” option in the **`wait-time`** step specifies what the caller hears if the caller cannot be connected to the specified source. Or, when the call is waiting in queue, the “then” option specifies what the caller hears if the call is not answered in 20 seconds. In this example, if the call is not answered in 20 seconds, the caller hears system music until a subsequent **`announcement`**, **`busy`**, **`collect`**, **`converse-on`**, **`disconnect`** or **`wait-time`** step is encountered.

You can specify **`music`** (system music), **`ringback`**, **`silence`**, or **`continue`** for the **`then`** option. When continue is specified, the caller continues to hear the alternative audio or music source until it is replaced by a subsequent vector step regardless of the time specified in the **`wait-time`** step.

You can use alternate audio or music sources in vector loops to provide continuous audible feedback as shown in the following example vector steps.

Call delay with continuous audible feedback

```
1. ...
2. ...
3. ...
4. wait-time 30 secs hearing 55558 then continue
5. route-to number 913034532212 with cov n
6. goto step 4 if unconditionally
```

In the example shown above, a look-ahead call attempt is placed every 30 seconds on behalf of the caller. If extension 55558 is a long, barge-in, repeating announcement, the caller hears announcement 55558 all the way to the end without the announcement being restarted each time vector processing returns to step 4.

Multiple music sources on hold

This feature uses the tenant partitioning tenant number (TN) to determine which music source to use when a call is put on hold. You can assign a different music source to each possible TN.

For more information, about tenant partitioning, see *Administrator Guide for Avaya Communication Manager*, 555-233-506.

The COR setting of the station or extension that puts the call on hold determines whether music-on-hold is applied. The TN that is assigned to the destination extension number determines the music source. You assign a music source number to the TN on the Tenant form. The physical location (port) of the music source is assigned on the Music Sources form.

The TN that is assigned to the active VDN on the VDN form determines the music source that is used. During vectoring, a **`wait hearing music`** command attaches the vector delay music source that is defined by the TN for the active VDN. The Multiple Music Sources for Vector Delay, which is a **`wait hearing extension then...`** command, applies the vector delay source that is defined by the **Announcements** or **Audio Sources** extension regardless of the TN that is assigned to the VDN.

Note:

> The TN administered for extensions on the **Announcement or Audio Sources** form applies only to direct calls to the announcement extension. For these calls, the announcement or music source assigned to the TN is what the caller hears.

During vector processing, if the **`converse`** vector command connects the call to an agent when the call remains under vector control and the agent puts the call on hold, the active VDN applies music-on-hold.

When a vector routes a call to another destination by a **`queue`**, **`check`**, **`route-to`**, or **`messaging split`** command, the switch uses the TN of the last active VDN to determine the music source for music-on-hold.

In ACD systems without vectoring and where music-on-hold applies, the TN assigned to the called hunt group extension determines which music source callers hear while in queue or on hold.

busy command

A busy tone and subsequent termination of vector processing are produced using the **`busy`** command. An exception to this occurs on CO trunks where answer supervision has not been sent. Callers on such trunks do not hear the busy tone from the switch. Instead, these callers continue to hear ringback from the CO. The **`busy`** command eventually times out and drops the call after 45 seconds. With ISDN PRI, busy tone can be provided from the network switch.

You might want to force a busy tone to process a call that arrives at a time when there are a large number of calls queued in the main split, or when the contact center is out of service or closed.

An example vector that demonstrates the **`busy`** command is shown below.

Busy command example

```
1. goto step 6 if calls-queued 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
```

In the example vector shown above, the **`goto step`** command in step 1 sends call control to **`busy`** in step 6 if the conditions in the former command are met. Specifically, if the number of calls that are queued at a high priority is greater than 30, the **`busy`** command is accessed.

disconnect command

The **`disconnect`** command forcibly disconnects a call with an optional announcement. Any previously established call treatment ends when the **`disconnect`** command is executed, and the call is removed from vector processing and from the switch.

If the call is connected to a station while the announcement is playing, the announcement stops and the caller hears ringback. Also, because vector processing stops when the call connects to a station, the disconnect portion of the command is not processed.

When the **`disconnect`** command includes an announcement, the switch sends answer supervision (if it was not already sent) just before the announcement plays.

When the **`disconnect`** command does not include an announcement, the switch sends answer supervision before it disconnects a call.

Note:

Answer supervision is not sent for ISDN trunks.

An example of the **`disconnect`** command is shown below.

Call disconnect example

```
disconnect after announcement 2918 (‘‘Today has
     been declared a snow day.  Please report for work tomorrow
     at 8 P.M.’’)
```

In this example, the caller is provided with sufficient information to meet their needs, so that no further interaction is required.

Routing commands

Basic Call Vectoring includes vectoring commands that enable you to route telephone calls.

queue-to split and check split commands

Calls that come in to the Call Vectoring system can be queued to a maximum of three ACD splits. Two commands are used to queue calls to splits.

The **`queue-to split`** command queues a call unconditionally. The command sends a call to a split and assigns a queuing priority level to the call in case all agents are busy.

The **`check split`** command conditionally checks the status of a split for possible termination of the call to that split. The command either connects the call to an agent in the split or puts the call into the split's queue at the specified priority level if the condition specified as part of the command is met.

Multiple split queuing

The term *multiple split queuing* refers to the queuing of a call to more than one split at the same time. The following example vector shows this process.

Multiple split queuing example

```
1. goto step 4 if calls-queued in split 1 pri l >= 10
2. queue-to split 1 pri t
3. wait-time 12 seconds hearing ringback
4. check split 2 pri m if calls-queued < 5
5. check split 3 pri m if calls-queued < 5
6. announcement 3001
7. wait-time 50 secs hearing music
8. goto step 4 if unconditionally
```

To avoid completing vector processing without queuing the call to a split, it is always good practice to check a split's queue before queuing to that split. If the queue is full, alternate treatment such as queuing to an alternate split should be provided. In this vector, if the main split's queue (which has 10 queue slots) is full, the **`goto step`** command in step 1 skips the main split and goes directly to step 4 to check the backup splits. Although calls are queued in step 2 at a top priority, a low priority is specified in step 1 so that calls in queue at all priority levels are counted. If there are 10 or fewer calls in the main split, control is passed to step 2, where the **`queue-to split`** command queues the call to split 1. Once the call is queued, vector processing continues at the next step.

Step 4 contains a **`check split`** command. If the call is not answered by the time step 4 is reached, the **`check split`** in the step attempts to queue the call to a second split. Specifically, the command first determines whether there are fewer than five calls that are queued to split 2. If so, the command then attempts to connect the call to an agent in the split. If such a connection cannot be made, the command puts the call into the split's queue at the specified priority level. Vector processing then continues at the next step. If there are five or more calls queued to split 2, the command fails and vector processing continues at step 5.

Step 5 contains another **`check split`** command and, accordingly, the process described in the previous paragraph is repeated, with one difference: the queuing attempt is made to split 3 instead of to split 2.

Except for the condition check, the circumstances under which the **check split** command cannot queue a call are identical to those for the **queue-to split** command.

Finally, note that whenever a call is queued to a backup split, the call remains queued to the main split and/or to another backup split if it was already queued to either or both of these splits. Once the call is answered in a split to which it is queued, the call is automatically removed from all the other splits to which it is also queued.

Note:

The **check split**, **queue-to-split**, and converse-on commands can access only those splits that are vector-controlled. A split is considered to be vector-controlled if **yes** is entered in the Vector field of the Hunt Group form.

Option with the VDN as the coverage point

The Vector Directory Number (VDN) can be used as the last point in a coverage path. This capability allows the call to first go to coverage and then to be processed by Call Vectoring and/or Call Prompting. The capability also allows you to assign AUDIX to a vector-controlled hunt group and to therefore enable access to these servers using a **queue-to split** or **check split** command. The result of all this is that call handling flexibility is enhanced.

The following example shows a vector, for which the VDN serves as a final coverage point, that allows the caller to leave a recorded message.

Leaving recorded messages (VDN as the coverage point option)

```
VDN 1 (used in a coverage path)
Vector 1
    1. goto step 7 if time-of-day is mon 8:01 to fri 17:00
    2. goto step 13 if staffed-agents in split 10 < 1
    3. queue-to split 10 pri 1 (AUDIX split)
    4. wait-time 20 seconds hearing ringback
    5. announcement 1000 (‘‘Please wait for voice
       mail to take your message.’’)
    6. goto step 4 if unconditionally
    7. goto step 2 if staffed-agents in split 20 < 1
    8. queue-to split 20 pri 1 (audix split)
    9. wait-time 12 seconds hearing ringback
   10. announcement 1005 (‘‘Please wait for an attendant
       to take your message.’’)
   11. wait-time 50 seconds hearing music
   12. goto step 10 if unconditionally
   13. disconnect after announcement 1008 (‘‘We cannot
       take a message at this time.  Please call back tomorrow.’’)
```

In steps 3 and 8 of the vector example shown above, the caller is given the option of leaving a recorded message, but the **queue-to split** command instead of the **messaging split** command is used in each case. Thus, the call is actually queued to the AUDIX split.

However, a **messaging split** command does not queue the call to the split. Instead, if it is successful, it connects the caller to the split so the caller can leave a message for the specified extension. However, termination to the split may turn out to be unsuccessful due to a factor that cannot be checked by vector processing. For example, the AUDIX link might not be functioning, or all AUDIX ports might be out of service.

As a result of the queuing process, a wait-announcement loop can be included after each **queue-to split** step, and the appropriate loop can then be executed until the call is actually terminated to either an AUDIX voice port or to an available message service agent. In this vector, steps 4 through 6 comprise the first wait-announcement loop, and steps 10 through 12 comprise the second such loop.

messaging split command

Basic Call Vectoring allows the caller to leave a message for the customer if the agents at the customer site are not available to take telephone calls. This is done with the help of the **messaging split** command. The following example illustrates use of the **messaging split** command.

Leaving recorded message

```
1. goto step 8 if time-of-day is all 16:30 to all 7:30
2. goto step 10 if calls-queued in split 47 pri l >= 20
3. queue-to split 47 pri m
4. wait-time 12 secs hearing ringback
5. announcement 4001
6. wait-time 60 secs hearing music
7. goto step 5 if unconditionally
8. announcement 4111(‘‘We’re sorry, our office
   is closed.  If you’d like to leave a message, please
   do so after the tone.  Otherwise, please call back
   weekdays between 7:30 A.M. and 4:30 P.M.  Thank you.’’)
9. goto step 11 if unconditionally
10.announcement 4222 (“We’re sorry, all of our agents are busy, please leave
   a message after the tone and we will return your call.”)
11. messaging split 18 for extension 2000
12. disconnect after announcement 4333 (‘‘We’re sorry, we are
    unable to take your message at this time.  Please call back at your
    convenience weekdays between 7:30 A.M. and 4:30 P.M. Thank you.’’)
13. busy
```

In this vector, the **goto step** command in step 1 checks to see if the office is open, and branches to step 8 if the office is closed. This is done to accommodate calls that are made during nonworking hours, when there are no agents available to take telephone calls. Accordingly, step 8 provides the caller with an appropriate announcement and an opportunity to leave a recorded message.

Step 2 checks to see if split 47’s queue (which has 20 queue slots) is full, and branches to step 10 if it is. Steps 3 to 7 queue the call to split 47 and then give audible feedback to the caller.

If the caller chooses to leave a message, the **`messaging split`** command in step 11 is executed. Split 18 in the command is the AUDIX split. Extension 2000 is the mailbox for split 47 (from step 2).

Upon execution of the **`messaging split`** command, an attempt is made to connect the caller to AUDIX so that he or she can leave a recorded message. If the split queue is full, or if the AUDIX link is not functioning, termination to AUDIX is unsuccessful, and vector processing continues at the next vector step. As is the case here, this step usually contains an announcement that provides the caller with the appropriate apology and subsequent directives. If the caller is successfully connected to AUDIX, vector processing terminates, and a message can be left for the specified mailbox. In this case, the mailbox is 2000.

Finally, if the supervisor or a group of agents has an Automatic Message Waiting (AMW) lamp for the mailbox used, and if the lamp lights, the relevant party, upon returning, knows that a caller has left an AUDIX message.

route-to number command

The **`route-to number`** command can be used to route calls to a vector-programmed number.

Interflow routing

Calls can be queued to a maximum of three splits. Calls can also be routed to a programmed number in the vector using a process that is known as interflow.

Note:

This feature is not related to the Look-Ahead Interflow feature. The Look-Ahead Interflow feature is not supported on Category B products.

Interflow allows calls that are directed or redirected to one split to be redirected to an internal or an external destination. For Basic Call Vectoring, this destination is represented by a number programmed in the vector. The number is always included in the **`route-to number`** command and it may represent any of the following destinations:

- Attendant or attendant queue
- Local extension
- Remote (UDP) extension
- External number
- VDN

When the **`route-to number`** command is used to chain multiple vectors together to enhance processing capabilities, the following events occur:

1. Vector processing continues at the first step in the vector assigned to the routed-to VDN.
2. The call (if queued) is dequeued.

3. Wait treatment (if any) is disabled.
4. Processing then continues in the receiving vector at step 1.

An example of vectors that demonstrate call interflow is shown below.

Call interflow example

```
VDN (extension=1000   name=‘‘Billing Service’’  vector=55)
Vector 55:
      1. announcement 3001
      2. goto step 8 if oldest call-wait in split 1 pri l > 120
      3. goto step 8 if calls-queued in split 1 pri l > 10
      4. queue-to split 1 pri t
      5. wait-time 50 seconds hearing music
      6. announcement 3002
      7. goto step 5 if unconditionally
      8. route-to number 2020 with cov n if unconditionally

  VDN (extension=2020 name=‘‘Message Service’’  vector=100)
  Vector 100:
      1. announcement 3900 (‘‘We’re sorry, all our
         agents are busy. Please leave a message. Thank you.’’)
      2. messaging split 18 for extension 3000
      3. disconnect after announcement 2505 (‘‘We cannot
       take a message at this time. Please call back tomorrow.’’)
```

In the first vector, a branch is made to step 8 from step 2 if the condition in the latter step (**`oldest call-wait in split 1 > 120 seconds`**) is true. If the condition is false, a branch is made to step 8 from step 3 if the condition in the latter step (**`calls-queued in split 1 > 10`**) is true. If that condition is also false, the call is queued (step 4), and a wait-announcement loop becomes effective (steps 5 through 7).

If a successful branch to step 8 is made from step 2, the **`route-to number`** command is executed. The destination number in this particular command, 2020, is a VDN. Accordingly, vector processing terminates in the first vector and begins at the first step of the second vector, to which the VDN points.

Once processing control is passed to the second vector, the caller is provided with the appropriate announcement (step 1). Thereafter, upon execution of the **`messaging split`** command in step 2, the system attempts to either queue the call to the message service split or else terminate the call to a message service agent or to an AUDIX voice port. If one of these attempts succeeds, the caller can leave a message. If none of the attempts succeed, the command fails, and vector processing continues at the next vector command. There is usually an announcement that explains to the caller that the necessary connection could not be made.

Branching/Programming commands

Basic Call Vectoring provides programming methods that can be used within a vector either to create branching patterns in call processing flows, or stop vector processing.

Branching/programming commands include:

- goto step and goto vector commands on page 68
- stop command on page 69

goto step and goto vector commands

The **goto step** and **goto vector** commands can be written to use unconditional branching or conditional branching.

The **goto** commands can also be used to connect multiple VDNs. If this command is used in a vector stem to connect to a different VDN, the following events occur:

1. Vector processing continues at the first step in the branched-to vector.
2. Call (if queued) remains in queue.
3. Wait treatment (if any) is continued.
4. Processing then continues in the receiving vector at step 1.

Unconditional branching

Unconditional branching is a method that always passes control from the current vector step to either a preceding or subsequent vector step or to another vector. This type of branching is enabled via the **goto step** and **goto vector** commands, each with a condition of **unconditionally** assigned.

An example of a vector in which unconditional branching is used is shown below.

Unconditional branching example

```
1. goto step 8 if calls-queued in split 3 pri m > 10
2. queue-to split 3 pri m
3. wait-time 12 seconds hearing ringback
4. announcement 3001
5. wait-time 30 seconds hearing music
6. announcement 3002
7. goto step 5 if unconditionally
8. busy
```

In the example shown above, the unconditional branch statement in step 7 establishes a loop between steps 5 through 7. Vector processing within the loop terminates when:

- an agent answers the call
- the system recognizes that the caller abandoned the call

Conditional branching

Conditional branching is a method that conditionally passes control from the current vector step to either a preceding or subsequent vector step or to a different vector. This type of branching is enabled via the **`goto step`** and **`goto vector`** commands, each with one of the following conditions assigned and tested: **`available-agents`**, **`staffed-agents`**, **`calls-queued`**, **`oldest call-waiting`**, or **`time-of-day`**. If the command condition is not met, control is passed to the step that follows.

An example vector in which conditional branching is implemented is shown below.

Conditional branching example

```
1. goto vector 100 if time-of-day is all 17:00 to all 8:00
2. goto vector 200 if time-of-day is fri 17:00 to mon 8:00
3. goto step 8 if calls-queued in split 1 pri l > 5
4. queue-to split 1 pri l
5. announcement 4000
6. wait-time 60 seconds hearing ringback
7. goto step 5 if unconditionally
8. busy
```

In the example shown above, a conditional branch test statement appears in steps 1, 2, and 3. If the call is placed during nonbusiness hours, which are defined as 5:00 p.m. to 8:00 a.m., the **`goto vector`** command in step 1 routes the call to vector 100. However, if the call is placed during business hours, control is passed to step 2, where the **`goto vector`** command there checks whether the call is placed during the weekend. If this is the case, the call is routed to vector 200. If not, control is passed to step 3, where the **`goto step`** command checks for the number of calls that are queued to the main split. If the number of calls is greater than five, control is passed to **`busy`** in step 8. If the number of calls is five or less, the call is queued (step 4). Thereafter, an announcement-wait cycle (steps 5 through 7) is implemented until an agent answers the call or the call is abandoned.

stop command

The **`stop`** command halts the processing of any subsequent vector steps. If a call is not queued when vector processing stops, the call is dropped and tracked as an “abandon” by BCMS. After the **`stop`** command is processed, any calls that are already queued remain queued, and any wait treatment is continued. Wait treatments include silence, ringback, system music, or alternate audio or music source.

A vector example that uses the **`stop`** command is shown below.

Stopping vector processing

```
1. goto step 6 if calls-queued in split 21 pri m > 10
2. queue-to split 21 pri m
3. announcement 4000
4. wait-time 30 seconds hearing ringback
5. stop
6. busy
```

If the **`stop`** command is reached, the queued caller continues to hear ringback. Also, if the **`stop`** command in step 5 is executed, step 6 is not executed immediately thereafter. The latter step can be executed only if the **`goto`** command in step 1 succeeds.

A vector will automatically stop processing when:

- The last step vector step is processed
- 1000 vector steps have been processed

Considerations

You should keep the following considerations in mind when working with Basic Call Vectoring:

- Make the split queues large enough so that all incoming calls queue and are not dropped. If a queue is too small, a **`queue-to split`** or a **`check split`** command might fail to queue a call due to a lack of available queue slots. Accordingly, it is also always a good practice to include in the vector a step that checks a split's queue before queuing occurs and a corresponding step that provides alternate treatment if the queue is full. To check the queue size, you can use a **`goto`** command (for example, **`goto Step 5 if calls-queued in split 20 pri l > 30`**). The alternate treatment, which, if needed, is usually accessed by the **`goto`** command that checks the queue size, can queue the call to a backup split, make an unconditional Look-Ahead Interflow attempt, provide a busy signal, etc.
- A default treatment or a **`route-to`** destination step should be supplied after a **`route-to`** command in case the first destination is unavailable.
- Calls should not be queued to an unstaffed split (unless this is intended by the customer) without some alternate treatment.
- Interflow calls should not be permitted to interflow back and forth between a remote switch vector and a local switch. This process could cause a single call to use up all available trunks.
- After an announcement is provided, the audible feedback (such as music) should be re-attached.
- For ease-of-use purposes, each specific vector function or operation should be included in a separate vector and linked via one or more **`goto vector`** commands.
- In creating a vector, commands can be chosen and arranged in a manner such that answer supervision is delayed as long as possible. This should be done to keep down the service cost.
- The caller should always be provided with initial feedback (usually ringback).
- Direct agent calls merit special attention because such calls can affect call queuing. Although direct agent calls take up a queue slot, they are not always reported as using such a slot on the BCMS reports (discussed in Appendix E: Call Vectoring and BCMS interactions on page 219). For example, a direct agent call is never counted toward the total of queued calls within a split (that is, the **`calls-queued`** test condition has no effect on this type of call).

Chapter 5: Call Prompting

Call Prompting provides flexible call handling that is based on information that is collected from a calling party. This information is in the form of dialed digits that originate from an internal or external touch-tone telephone or from an internal rotary telephone that is on the same switch as the vector. Call Prompting allows for the temporary transfer of call management control to the caller.

Call Prompting can be used in various applications so that calls can be handled with more flexibility.

This chapter includes the following topics:

Command set

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

Call Prompting command set

Command category	Action taken	Command
Information collection	Collect information from the calling party, from the public network in an ISDN SETUP message.	**`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

Before the switch can accept the touch-tone digits that are entered by a caller, the switch must be equipped with a “collection resource.” The resource used for collecting and interpreting touch-tone digits is a unit of hardware called a Touch-Tone Receiver (TTR). These 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 two 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 Account Team recommends the appropriate number of TTRs based on two 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 timeout 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:

The Call Prompting interdigit timeout can be administered for any number of seconds from 4 to 10. This value is administered on the Feature-Related System Parameters form.

Provisions for users of rotary telephones are illustrated in the vector scripts in this chapter.

Call Prompting digit entry — collect digits command

The touch-tone digits that are entered by a Call Prompting user are collected by the **`collect digits`** command. This command allows the system to collect up to 24 digits from a touch-tone telephone. Sixteen of these digits may be collected immediately, while any remaining digits are stored as dial-ahead digits, which are explained later in this chapter.

Call Prompting allows some flexibility in entering digits. Specifically, the caller can:

- Remove incorrect digits strings
- Enter variable-length digit strings
- Enter dial-ahead digits.

The following sections explain these processes.

Removing incorrect digit strings

An announcement that requests the caller to enter digits can be included in call treatment. As an option, the announcement can instruct the caller to enter an asterisk (*) if he or she enters incorrect data.

When the caller enters a “*”, the following happens:

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

   Note:

   Also deleted are any dial-ahead digits that are entered and that do not exceed the maximum digit count of 24. (Dial-ahead digits are explained later in this chapter.)

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

Once the caller enters an asterisk, the caller can reenter digits for processing.

Entering variable-length digit strings

The maximum number of digits that are requested from the caller must be specified in the administration of the **`collect digits`** command. In some cases, the caller might be permitted 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.

Call Prompting allows for variable-length digit strings by providing an end-of-dialing indicator in the form of the pound sign (#). The pound sign is used to end any digit string that is entered by the caller, and it does the following:

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

Whenever the caller is permitted to enter a variable-length digit string, the announcement portion of the **`collect digits`** command should specify the largest possible number of digits that can be entered. Accordingly, each **`collect digits`** command should 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 should 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 should 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 terminates the command, and any digits collected prior to the timeout are available for subsequent vector processing.

A common application involving the entering of variable-length digit strings allows the user to dial either the number for the attendant or an extension to reach the desired 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 should dial “0#.” However, if the user chooses to dial a 3-digit extension, the user should dial, for example, “748” and not “748#.” Since the maximum number of digits that can be dialed in this case is three, dialing “748#” would cause “#” 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 4, “#” is not saved as a dial-ahead digit since it is the fourth of four digits that can be entered in this case.

Entering 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. See Collecting Digits on the switch on page 167 for more information. These “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. Call Prompting also provides some additional functions that involve digit processing. These functions include the following:

- Treating digits as a destination
- Using digits to collect branching information
- Using digits to select options
- Displaying digits on the agent’s set

These functions are illustrated in the following sections.

Treating digits as a destination

Call Prompting allows you to route calls according to the digits that are 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 that is indicated by the digits that are processed by the most recent collect digits command.

The digits can represent any of the following destinations:

- Internal (local) extension, for example, split/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) feature access code (FAC) followed by a public network number, for example, 7 digit 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 he or she would like to reach (step 2). The extension in this vector may contain up to 5 digits. The vector collects the digits and then 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 might be in use. As a result, when you are collecting digits from a caller, you should avoid starting your main vector with a **`collect digits`** command, since the caller in this case receives no audible feedback if he or she has to wait for a TTR to become available. Accordingly, it is a good practice to include some treatment, for example, **`wait-time 0 seconds hearing ringback`**, before the initial **`collect digits`** step.

In addition, if calls are likely to be transferred to this vector, a wait-time step of sufficient length is recommended before the collect step to allow the transferring party enough 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
```

With Category B products, if you are comparing collected digits, you can only use “=” (equal). So, callers entering the digits 10111 are routed to vector 8; callers entering the digits 11111 are routed to vector 9; and callers entering the digits 12111 are routed to vector 10.

```
 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
```

Using digits to select options

Call Prompting makes it possible to provide a menu of options that the caller can use to satisfy his or her information needs. The caller selects 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’d 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 user is asked to enter either 1 or 2, depending on the service he or she uses. If one of these digits is entered, the appropriate one of the next two steps (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 the agent’s set

A CALLR-INFO button can be included at the agents’ display stations to help process calls that are serviced by the Call Prompting feature. However, if the agent has a two-line display set, and the display is in normal or inspect mode, the collected digits are automatically displayed on the second line. These digits remain on this line until they are overwritten, even after the call is released by the agent. On the other hand, for other display sets, the agent must press the CALLR-INFO button to display the collected digits.

It may be beneficial to install the CALLR-INFO button if you want to expedite calls by reducing the amount of time agents spend on the telephone. For example, the button could be set up to collect specific information such as a customer account number before the call is answered by the agent, thus eliminating the need for the agent to ask for this information.

The CALLR-INFO button displays information in the following format:

```
x = Info: 1234567890
```

where:

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

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

Assume that digits have been collected by Call Prompting. If the agent presses the CALLR-INFO button when the call rings at the agent station 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 (if available) that is 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 that was presented earlier in this section. Only those digits that were 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 (if available) that is associated with the button flashes denial.

One or more events may occur during a successful execution after the button is pushed. These 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 these events occur, the following takes place:

- The status lamp (if available) that is associated with the button is turned off.
- The display is updated as previously described.

Note:

If the agent needs to display the collected digits again, the CALLR-INFO button can be pressed again to repeat the operation that is described in this section, provided that the agent is active on the call or the call is still ringing. Also, the agent can flip between the collected digits and the ICI by alternately pressing the CALLR-INFO and NORMAL buttons.

Dial-ahead digits — collect digits command

Dial-ahead digits provide the caller with a means of bypassing unwanted announcement prompts on the way to acquiring the information or servicing he or she wants. These digits are available for use only by subsequent **`collect digits`** commands. The digits are never used by other vector commands that operate on digits, for example, **`route-to digits`**, and **`goto...if digits`**, until they are collected. These digits are not forwarded with interflowed calls. In addition, these digits are not displayed as part of the CALLR-INFO button operation until they are collected by a **`collect digits`** command.

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 switch storage limit of 24. Any additional digits are discarded until additional storage is made available by a subsequent **`collect digits`** command.

  Note:

  Any asterisk (*) and pound sign (#) digits that are dialed ahead count toward the 24 digit limit, as do any dial-ahead digits that are entered after the asterisk or pound sign digit.

- The TTR required by the user to collect digits is disconnected. This happens whenever 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 Call Prompting timeout occurs, during which time the caller has not dialed any additional digits, asterisks (*) or pound signs (#).
  - — Vector processing stops or is terminated.
  - — A successful or unsuccessful collect ced/cdpd step is encountered.

  Note:

  When the TTR is disconnected due to a **`route-to number`** step, all dial-ahead digits are discarded. This means that following a failed **`route-to`** step, a subsequent **`collect digits`** step always requires the user to enter digits.

Dial-ahead digit vector examples

The vectors shown in the following examples illustrate a situation where a caller can enter dial-ahead digits. In this case, the caller is required to have a touch-tone telephone. An alternative handling sequence should be programmed in case the caller has a rotary telephone or the caller does not dial a touch tone digit before the timeout period.

Step 2 of Vector 30 gives the caller two options, each of which provides different information. The caller is prompted to enter either 1 or 2, depending on what information he or she 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 (below), the caller is given three options that supplement the original option that was provided in Vector 30. The caller is prompted to enter either 3, 4, or 5, depending on what information he or she 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 (below) 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 at least two sets of options to get the information that he or she wants. Each option set is introduced by an announcement. However, because of the dial-ahead digit capability, the caller can bypass the announcements if he or she chooses. Thus, the caller could enter 1 and 5 within a matter of seconds to hear yesterday’s Southern Division scores.

The caller may enter digits while he or she 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, please press 6.  If you wish to
     hear today’s schedule of games in the Northern
     Division only, please press 7.  If you wish to hear
     today’s schedule of games in the Southern Division
     only, please 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
     (‘‘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 digits = 0
13.  announcement 301 (’’Entry not understood.  Please
     try again.‘‘)
14.goto step 10 if unconditionally
15.disconnect after announcement none
```

Considerations

You should keep the following considerations in mind when working with Call Prompting:

- To enter the digits requested via a **`collect digits`** command, outside callers must have a touch-tone telephone. For such callers using rotary dialing, a 10 second inter-digit timeout 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) should 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 timeout 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 via a **`route-to digits`** command, the **`route-to digits`** command fails, and vector processing continues at the next step, which should be a default treatment.
- It may be prudent to take steps in case a **`route-to attendant`** command fails, such as providing a disconnect announcement.
- From time to time, all of the system's touch-tone receivers might be in use. As a result, you should avoid starting your main vector with a **`collect digits`** command, since the caller on a DID or tie trunk in this case receives no audible feedback if he or she has to wait for a receiver to become available. Accordingly, it is a good practice to include some treatment (for example, a **`wait-time 0 seconds hearing ringback`** step) before the initial **`collect digits`** step.

Chapter 6: Attendant Vectoring

The Attendant Vectoring feature enables a set of commands that can be used to write call vectors for calls to be routed in non-contact center environments. When Attendant Vectoring is enabled, all attendant-seeking or “dial 0” calls are processed using the call vectors, not the normal attendant console call routing.

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 could be answered by an attendant, but it may also be covered to the voice mailbox of a night station. Training users to understand these different call routing options is something you should consider before using Attendant Vectoring.

If you use Attendant Vectoring and night service to route calls to a voice mail system, you can also use the Automatic Message Waiting (AMW) feature to notify after-hours personnel that there are messages in the night service station mailbox by assigning an AMW lamp on one or more backup telephones. When personnel see that there are new messages, they can check those messages after hours and act upon them as needed.

This chapter includes the following sections:

Command set

The following table lists the commands associated with Attendant Vectoring.

Attendant vectoring command set

Command category	Action taken	Command
Treatment		
	Play an announcement. Play a busy tone and stop vector processing. Disconnect the call. Delay with audible feedback of silence, ringback, system music, or alternate audio/music source.	**`announcement`** **`busy`** **`disconnect`** **`wait-time`**
Routing		

	Queue the call to an attendant group.
	Queue the call to an attendant extension.
	Queue the call to a hunt group.
	Route the call to a specific extension number.
		`queue-to attd-group`
		`queue-to attendant`
		`queue-to hunt-group`
		`route-to number`

Branching/programming		

	Go to a vector step.
	Go to another vector.
	Stop vector processing.
		`goto step`
		`goto vector`
		`stop`

Treatment commands

Attendant Vectoring allows use of several TREATMENT commands, including:

- announcement command *on page 91*
- busy command *on page 91*
- disconnect command *on page 91*
- wait-time command on page 91

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.

announcement command

Syntax: announcement <extension>

The usage for the **`announcement`** command is the same as in Basic Call Vectoring. See the Basic Call Vectoring chapter for details on using this command.

busy command

Syntax: busy

The usage for the **`busy`** command is the same as in Basic Call Vectoring. See the Basic Call Vectoring chapter for details on using this command.

disconnect command

Syntax: disconnect after announcement <extension>

The usage for the **`disconnect`** command is the same as in Basic Call Vectoring. See the Basic Call Vectoring chapter for details on using this command.

wait-time command

Syntax: wait-time <time> secs hearing <silence, ringback, music>

This use of the **`wait-time`** command was slightly modified for attendant vector usage. The **`i-silent`** treatment choice was removed because it does not pertain to attendant vectoring. The **`wait-time <seconds> secs hearing <extension> then <silence, ringback, music, continue>`** command was left unchanged. No other changes or attendant specific considerations apply, so these commands work as they do in Basic Call Vectoring.

Routing commands

Attendant Vectoring allows use of several ROUTING commands, including:

- queue-to attd-group command *on page 92*
- queue-to attendant command *on page 93*
- queue-to hunt-group command *on page 94*
- route-to number command on page 94

Note:

> A **`wait-time 0 secs hearing ringback`** step should be used to give immediate feedback to the caller. The **`queue-to`** command does not provide ringback until the call is actually ringing the attendant. The **`wait-time`** step should be implemented as the first vector step or as the step immediately before the **`queue-to`** step.

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.

queue-to attd-group command

Syntax: queue-to attd-group

The **`queue-to attd-group`** vectoring command is available only for attendant vectors. If an attendant group call is redirected to vector processing that queues the call to the attendant group, the group to which the call gets queued is determined by the TN assignment that is associated with the call. If an attendant in the group is available to take the call, it is terminated to the attendant, not queued, and vector processing terminates.

Attendant group based on tenant number

When attendant group calls are redirected to vector processing and are programmed to queue to the attendant group, the attendant group is the group that is designated for the call's associated tenant number.

If an attendant group call is redirected to vector processing that queues the call to the attendant group, the call is placed in the queue using the priority that is assigned for the call. Attendant queue priorities are assigned on a system-wide basis, not on an individual partition basis.

Attendant group queue

Calls that are queued to the attendant group by way of attendant vector processing are queued with the system-administered priority for the call. If an attempt is made to queue the call and it fails, the vector event for queue failure is logged.

As with other vector queue commands, vector processing continues with the next step following the **`queue-to attd-group`** command regardless of success or failure. The **`goto step if queue-fail`** command is provided for handling failure conditions. Otherwise, on success, announcements or other feedback can be applied while the call is in queue. Other than the provision of caller feedback, attendant queue functionality is unchanged. If no commands follow a successful queue step, the call is left in the queue with no feedback. If no commands follow a failed queue step, the call is dropped. Anytime the end of vector processing is reached without the call being placed in queue, it is dropped and an event is logged.

queue-to attendant command

Syntax: queue-to attendant <extension>

The **`queue-to attendant`** vectoring command is available only for attendant vectors. If an attendant group call is redirected to vector processing that queues the call to an individual attendant, the attendant to whom the call gets queued must be a member of the attendant group that is indicated by the TN assignment associated with the call. If the attendant is available to take the call, the call is terminated to the attendant, not queued, and vector processing terminates.

The success of this command depends on having individual attendant access. These calls are queued based on the priority that is assigned to individual attendant access calls.

Individual attendant queue

Calls that are queued to the individual attendant via attendant vector processing are queued with the system-administered priority for individual attendant access calls. If the indicated attendant is not a member of the associated attendant group, the command is considered failed and vector processing continues with the next vector step. If an attempt is made to queue the call and it fails, a vector event is logged.

As with other vector queue commands, vector processing continues with the next step following the **`queue-to attendant`** command regardless of success or failure. The **`goto step if queue-fail`** command is provided for handling failure conditions. Otherwise, on success, announcements or other feedback can be applied while the call is in the queue. If no commands follow a successful queue step, the call is left in the queue with no feedback. If no commands follow a failed queue step, the call is dropped. Anytime the end of vector processing is reached without the call being placed in queue, the call is dropped and an event is logged.

queue-to hunt-group command

Syntax: queue-to hunt-group <#> pri <l (low), m (medium), h (high), t (top)>

This vectoring command is available only for attendant vectors. However, it is the functional equivalent of the split queueing command. As such, a call can be queued to up to three hunt groups. If an attendant group call is redirected to vector processing that queues the call to a hunt group, the call is queued with the indicated priority. If a hunt group member is available to take the call, it is terminated to the member, not queued, and vector processing terminates. In order to use a hunt group in vectoring, it must be administered as a vector controlled group. However, it can be any type of hunt group, including UCD, ACD, and so forth.

Hunt group queue

Calls that are queued to a hunt group by way of attendant vector processing are queued with the indicated priority for the call. If an attempt is made to queue the call and it fails, a vector event is logged.

As with other vector queue commands, vector processing continues with the next step following the **`queue-to hunt-group`** command regardless of success or failure. The **`goto step if queue-fail`** command is provided for handling failure conditions. Otherwise, on success, announcements or other feedback can be applied while the call is in the queue. Since these hunt groups are required to be vector-controlled, announcements are provided by way of vectoring commands and hunt group-specific forced announcements do not apply. If no commands follow a successful queue step, the call is left in the queue with no feedback and vector processing terminates. If no commands follow a failed queue step, the call is dropped. Anytime the end of vector processing is reached without the call being placed in the queue, it is dropped.

route-to number command

Syntax: route-to <number> with cov <y, n> if <unconditionally>

This command is slightly modified from standard usage when used for attendant vectoring and **`unconditionally`** is the only available option. Existing choices allow routing with **`if unconditionally`**, **`digit`**, or **`name`**. Since digit comparison and interflow do not pertain to attendant vectoring, the options are not available. No other changes or attendant specific considerations apply. This command works as it does in standard usage. This command is provided by administration that is defined on the Console Parameters form. Therefore, call processing requirements are not needed.

Branching/programming commands

Attendant Vectoring allows use of several branching/ programming commands, including:

- goto step command *on page 95*
- goto vector command *on page 96*
- stop command on page 96

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

Syntax: goto step <step #> if time-of-day is <day><hour>:<minute> to <day><hour>:<minute>

This use of the **`goto step`** command is the same as in Basic Call Vectoring. See the Basic Call Vectoring chapter for details on using this command.

Syntax: goto step <step #> if <unconditionally>

This use of the **`goto step`** command is the same as in Basic Call Vectoring. See the Basic Call Vectoring chapter for details on using this command.

Syntax: goto step <step #> if queue-fail and goto vector <vector #> if queue-fail

These vectoring conditionals are available only for attendant vectors. Any time an attempt is made to queue a call and it cannot be queued, these commands can be used to direct vector processing. For attendant vectoring, there is no attempt to determine whether a call can be queued before attempting to do so. Therefore, one of these commands can be used to provide alternate processing when calls cannot be queued. Some examples of why calls can fail to queue are as follows, but this is not a complete list of the causes of failure:

- The queue is full
- The attendant group is in night service and there is no night console
- The individual attendant is not a member of the associated attendant group
- There were invalid multiple queue attempts. See Attendant Vectoring and multiple queueing on page 105 for more information

Failure to queue

The queue failure conditional is set following a queue command that fails to queue the call. It always indicates the result of the most recent queue command. If the failure conditional is set, vector processing is redirected as indicated.

goto vector command

Syntax: goto vector <vector #> if time-of-day is <day><hour>:<minute> to <day><hour>:<minute>

The use of the **`goto step`** command is the same as in Basic Call Vectoring. See the Basic Call Vectoring chapter for details on using this command.

Syntax: goto vector <vector #> if unconditionally

The use of the **`goto step`** command is the same as in Basic Call Vectoring. See the Basic Call Vectoring chapter for details on using this command.

stop command

The use of the **`stop`** command is the same as in Basic Call Vectoring. See the Basic Call Vectoring chapter for details on using this command.

Overview

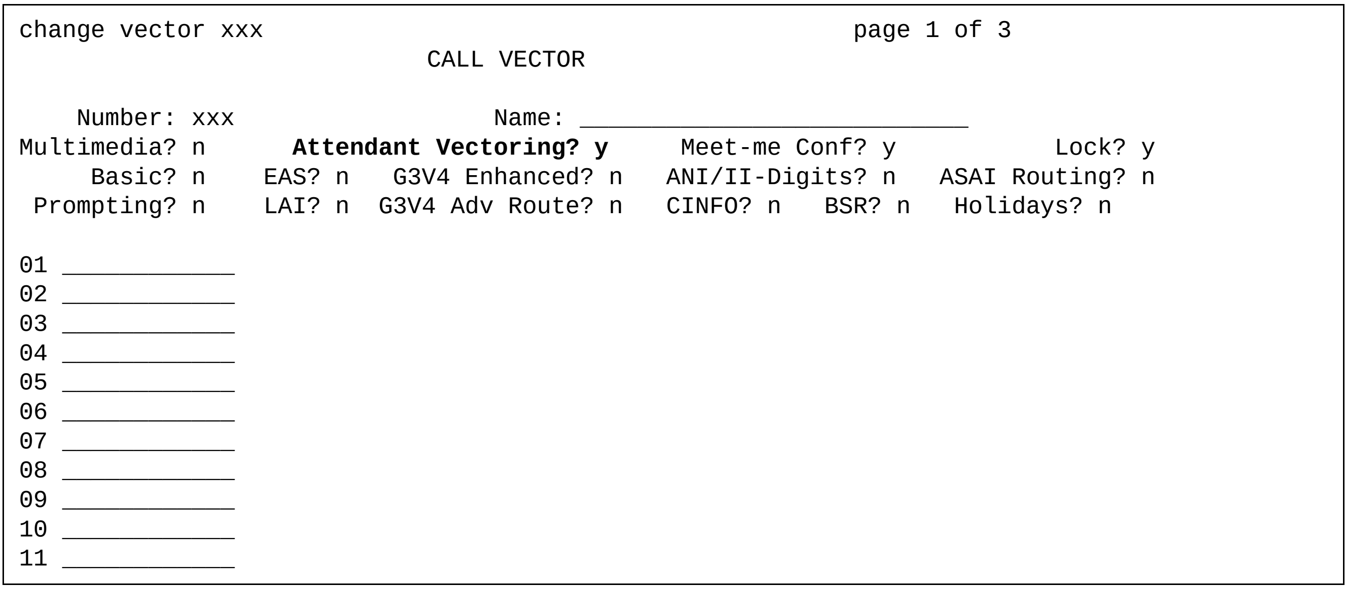
The Attendant Vectoring capability enables you to use certain vector commands in a non-contact center environment. For example applications of Attendant Vectoring see Call Vectoring applications on page 131.

Attendant Vectoring is available in nondistributed attendant environments and distributed attendant environments for IAS.

Vector form

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

Call Vector form



```
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 form. 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 form, the Attendant Vectoring field defaults to y, and it cannot be changed to n. When the Attendant Vectoring field on the Call Vector form 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 forms indicates attendant vectoring. When attendant vectoring is indicated for VDNs and vectors, all contact center-associated fields are not displayed.

Console Parameters form

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

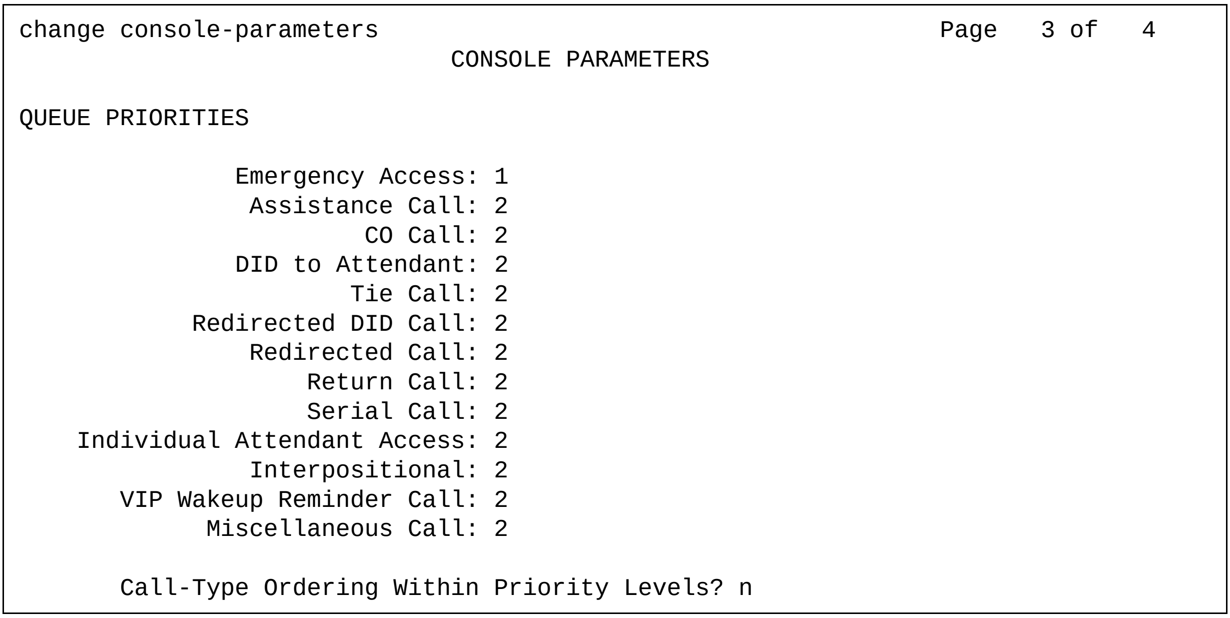
Console Parameters form (Page 1)

```
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 form (Page 2)

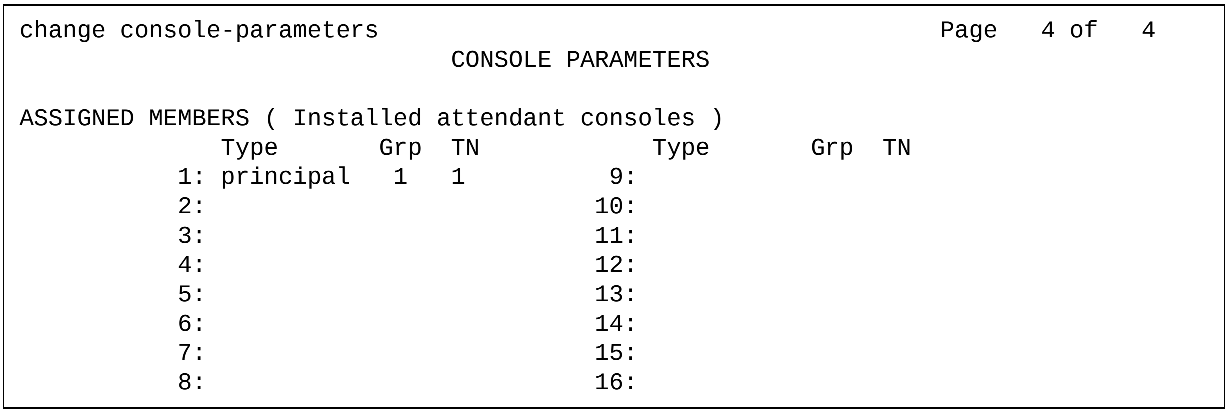
```
change console-parameters                                       Page   2 of   4
                              CONSOLE PARAMETERS

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 form (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 form (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:
```

TN assignments

Just as TN assignment determines to which attendant group calls are terminated, the TN assignment also determines to which VDN the 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. How the switch determines which party's TN to use in call scenarios is not changed by attendant vectoring and the VDN for the selected TN still applies.

For example, 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 form 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 form when partitioning is turned off. When it is turned on, the field is removed from the console form and the contents are automatically copied to TN 1.

Restrictions

No restrictions apply to attendant and nonattendant vectoring. For example, an attendant VDN can point to a nonattendant 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 nonattendant 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 nonattendant VDN is routed to an attendant VDN.

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 form. 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.

Hunt group queue

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.

Redirecting calls to attendant VDNs

Because it is not possible to apply vector commands or specialized administration to specific types of attendant group calls, the following can not be redirected to the attendant VDN:

- 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 calls. 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 calls. As with 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 calls. These reminder calls are still sent directly to the attendant group.
- Call Park time-out. These calls result in a conference (caller, principal, and attendant) and call vectoring does not allow conferenced calls to be vectored.
- Call Transfer time-out. These calls are controlled by the attendant return call timer and are processed as though they are attendant extended calls, in other words, actual attendant return calls.

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, they 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.

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:

Hunt group night destination – An attendant vectoring VDN can be assigned as a hunt group's night destination. Calls to that 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 form is y, the `night service destination` field is removed from the form. 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 could be in night service and the call would then terminate to the indicated night service destination. If the hunt group is accessed via the **`queue-to hunt-group`** command no night service applies.

LDN and trunk night destination – One or all trunk groups can be placed into night service and an attendant vectoring VDN can be assigned as the group's night service destination. If a night destination is assigned for LDN calls, it overrides (for LDN calls) the trunk group's night destination. 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 that is associated with the trunk group's TN. If, instead, the night service destination is explicitly assigned to a particular attendant vectoring VDN, it may or may not be the VDN that would have resulted had the night destination been the attendant group.

Tenant night destination – For tenant partitioning, each partition can be assigned a night destination. When tenant partitioning is turned off, local attendant group calls are sent to the LDN night destination. When partitioning is turned on, local attendant seeking calls are sent to the partition's night destination.

Trunk group incoming destination – The incoming destination can be an attendant vectoring VDN. As in trunk group night service, an assigned incoming destination to an attendant vector could 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, it should 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 should be given to routing these calls off-switch.

Auxiliary data for the following button assignments – In keeping with existing procedures, attendant VDNs will not be 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 that is associated with the button.
- Manual signaling. Rings the station that is 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 form if tenant partitioning is turned off or on the Tenant form 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. The `Allow VDN Override`? field will not be available so **`yes`** is assumed.

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 nonattendant vector can be assigned to an attendant VDN and an attendant vector can be assigned to a nonattendant VDN. Obviously, doing so is not recommended. Assigning an attendant vector to a nonattendant VDN severely restricts processing for basic call vectoring since only limited vectoring commands are available in attendant vectors. Assigning a nonattendant 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 nonattendant 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, they are removed from the queue (if applicable) and treated as new calls to vectoring, not continuations of vectoring.

Tenant partitioning assignments apply to attendant VDNs the same as they do for nonattendant VDNs. Therefore, care must be taken that a VDN assignment on the partitioning form 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.

Attendant Vectoring and multiple queueing

Calls can exist in only one type of queue, which can be an attendant group, and individual attendant, or a hunt queue, and cannot be moved from one queue to another. For example, if a call is queued to the attendant group and a subsequent command attempts to queue the call to an individual attendant or hunt group, it is considered a failed queue attempt.

Restrict queueing to only one type of queue

Once a call is queued to the attendant group, individual attendant, or hunt group, any attempt to queue the call to another type of queue is considered a failed queue attempt.

Multiple attempts to queue to attendant groups or individual attendants are also considered failed queue attempts. For example, if a call is queued to attendant X and a subsequent command attempts to queue the call to attendant Y, the second queue command fails.

Allow multiple priority queueing within hunt queues

Since hunt group queueing is based on the indicated priority, multiple queue attempts are valid. There is no limitation on the number of attempts to queue to a particular hunt group so long as the command changes the priority at which a call is to be queued. For example, a call can be queued at low priority and subsequently requeued at medium and/or high priority. However, a second attempt to queue a call at the same priority for which it was previously queued is considered a failed queue attempt. Hunt group queueing is the functional equivalent to split queueing. As such, calls can be queued to a maximum of three different hunt groups at the same time.

Once a call is queued to a hunt group, any subsequent attempt to queue with a different priority results in the call being requeued with the new priority. Any subsequent attempt to queue with the same priority at which the call is already queued is considered a failed queue attempt.

Allow multiple hunt group queueing

A call can be queued to a maximum of three different hunt groups. Once this maximum is reached, any subsequent attempt to queue a call to a different hunt group is considered a failed queue attempt.

Considerations

The main consideration with Attendant Vectoring is training users to understand that calls placed to an attendant console may not always be answered by a live operator. If users are instructed to dial an attendant VDN, the call could be answered by an attendant, but it may also be covered to the voice mailbox of a night station. Training users to understand these different call routing options is something you should consider before using Attendant Vectoring.

If you use Attendant Vectoring and night service to route calls to a voice mail system, you can also use the Automatic Message Waiting feature to notify after-hours personnel that there are messages in the night service station mailbox by assigning an AMW lamp on one or more backup telephones. When personnel see that there are new messages, they can check those messages after hours and act upon them as needed.

Chapter 7: Holiday Vectoring

Holiday Vectoring enables a set of commands that can be used to write call vectors for calls to be routed on holidays or any days when special processing is required.

This chapter gives you the information you need to use this vectoring option.

This chapter includes the following major topics:

Command set

The following table shows the commands that are available for use in Holiday Vectoring.

Holiday Vectoring command set

Command category	Action taken	Command
Branching/programming		

	Go to a vector step
	Go to a vector
		`goto step`
		`goto vector`

Branching/programming commands

Holiday Vectoring allows use of two branching/programming commands, including:

- goto step command *on page 108*
- goto vector command *on page 108*

The following sections detail the syntax that can be used for these commands and any information that is specific to their use in Holiday Vectoring.

goto step command

Syntax: goto step <step #> if holiday in table <table #>

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.

Syntax: goto step <step #> if holiday not-in table <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

Syntax: goto vector <vector #> if holiday in table <table #>

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.

Syntax: goto vector <vector #> if holiday not-in table <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.

Overview

Holiday Vectoring is an enhancement that simplifies vector writing for holidays. It is designed for customers who need to reroute or provide special handling for date-related calls on a regular basis.

This feature provides the user with the capability to administer ten different Holiday Tables, then use those tables to make vectoring decisions. Each table can contain up to 15 dates or date ranges. All of this can be done in advance to ensure seamless call routing over holidays when staffing is reduced or contact centers are closed.

When vector processing encounters a **`goto xxx if holiday in table #`** step, it determines if the current date and time qualifies as a holiday according to the given table. That information is then used to decide whether the goto condition is true or false, and therefore, whether to **`goto`** the given step or vector or not. The date and time match is done at the time that the call is in vector processing. It is done just like time-of-day routing. This means that it is checking the system date and time on the PPN, rather than the local port network time on the EPN.

The Holiday Vectoring feature 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 semiannual sale.

This feature was developed in response to customer needs, especially for some customers who may have as many as 30 bank holidays to administer throughout the year. Holiday Vectoring streamlines vectoring tasks and ensures seamless operation over holiday (or special-event) periods.

Without this feature, contact center administrators had to write special vectors for each holiday or other special date-related circumstances, and make sure that these vectors were administered at the appropriate times. In some cases, administrators were required to go to work on holidays just to administer vectors. This feature was developed in response to customer needs, especially for some customers who may have as many as 30 bank holidays to administer throughout the year.

Administering Holiday Vectoring

This section gives you step-by-step instructions on setting up Holiday Tables and writing vectors to include Holiday Vectoring.

Enabling Holiday Vectoring

The Holiday Vectoring customer option can be enabled if:

- The switch software version is R9.1 or greater, and
- Either Vectoring (Basic) or Attendant Vectoring is enabled.

On the Customer Options Form, the Vectoring (Holidays) field should be set to y. If the feature is not enabled, contact your Avaya customer support or authorized representative to have the feature enabled.

Setting up a Holiday Table

This section describes how to set up a Holiday Table before adding to a vector.

Holiday Table command syntax

This section describes the syntax of each Holiday Vectoring command.

Syntax: 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 and give the number of a blank table. For example, change holiday-table 9, where table 9 has not been used to define holidays.

Syntax: display holiday-table x

This command allows you to display the entries in a Holiday Table.

Syntax: list holiday-table

This command lists all of the Holiday Tables.

Syntax: list usage holiday-table x

This command lists all vector steps that refer to the selected Holiday Table.

Using the Holiday Table commands

After ensuring that Holiday Vectoring is enabled on the Customer Options form, enter **`change holiday-table 1`**.

On the Holiday Table Form, 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               12    31                Christmas
    01    01  00   00      01    01  10   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. In the example above, two entries were made, one for each calendar year.

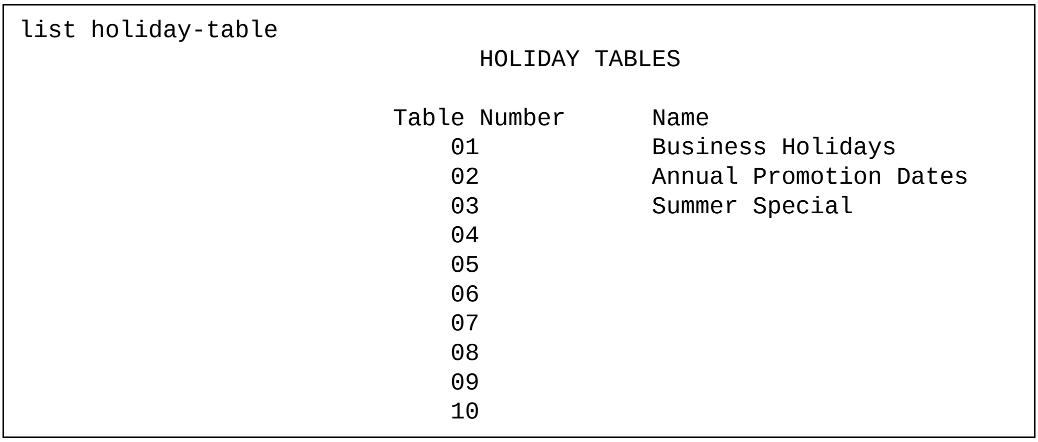
The Holiday Table Form can be used for entering individual holidays or holiday ranges. The following rules apply to entering dates on this form:

- If a day is entered, the corresponding month must be entered.
- If a month is entered, the corresponding day must be entered.
- If an hour is entered, the corresponding minute must be entered.
- If a minute is entered, the corresponding hour must be entered.
- If an hour and minute is entered, the corresponding month and day must be entered.
- If a month and day is entered, the corresponding hour and minute is not required.
- If an end month and day is entered, the corresponding start month and day must be entered.
- If a start month and day is entered, the corresponding end month and day is not required.
- To enter an individual holiday, enter a start month and day, but do not enter an end month and day.
- To enter a holiday range, enter both a start month and day and an end month and day.

- The start month, day, hour, and minute must be less than or equal to the end month, day, hour, minute.
- The description field is an alpha-numeric field that is used for identification.

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 the 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
```

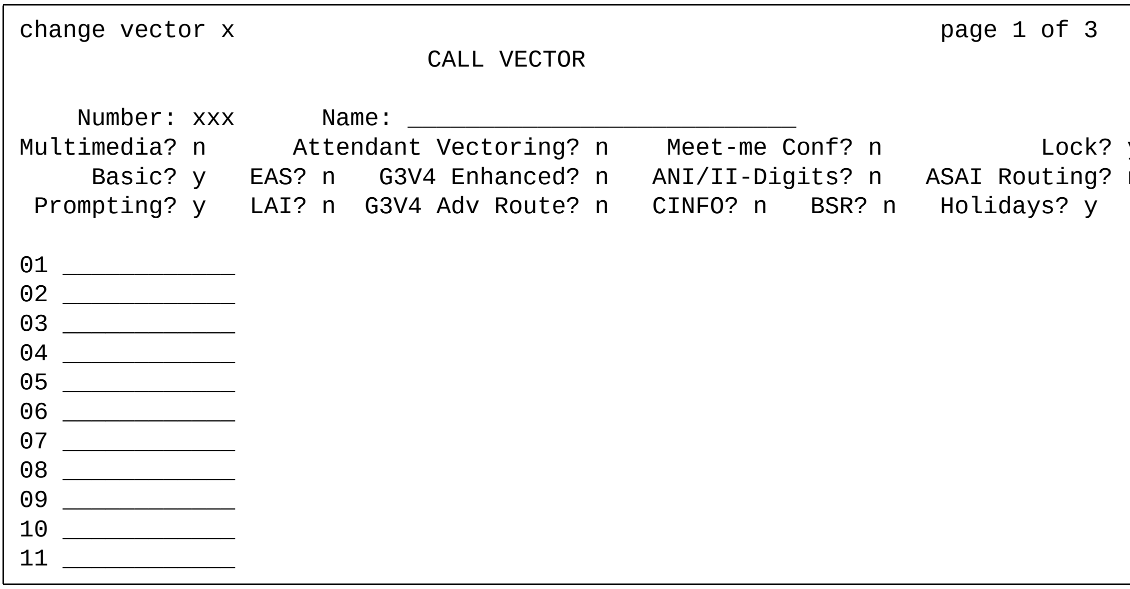
Changing vector processing for holidays

After administering the holiday tables, add or change vector processing for those holidays.

On the command line, enter **`change vector`** ***x*** (where ***x*** is the vector number). The Call Vector form contains a display-only field that indicates that Holiday Vectoring is enabled. On the Call Vector form, customers can enter a new goto conditional for the holidays.

When Holiday Vectoring is optioned, a field on the Vector form identifies if the vector on which you are currently working is a Holiday Vectoring vector, as shown in the following example.

Call Vector form

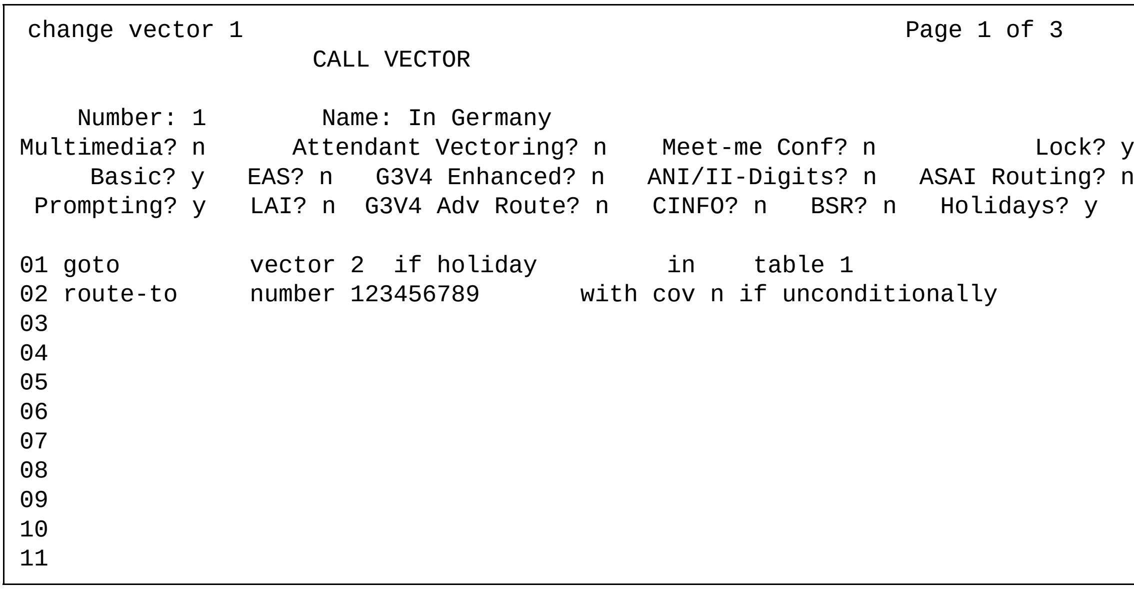


```
change vector x                                                    page 1 of 3
                            CALL VECTOR

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

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

The Holiday Vectoring field is a display-only field and appears only when Holiday Vectoring is enabled on the Customer Options form. If either Basic Vectoring or Attendant Vectoring are set to y, then the Holiday Vectoring field can be set to y.

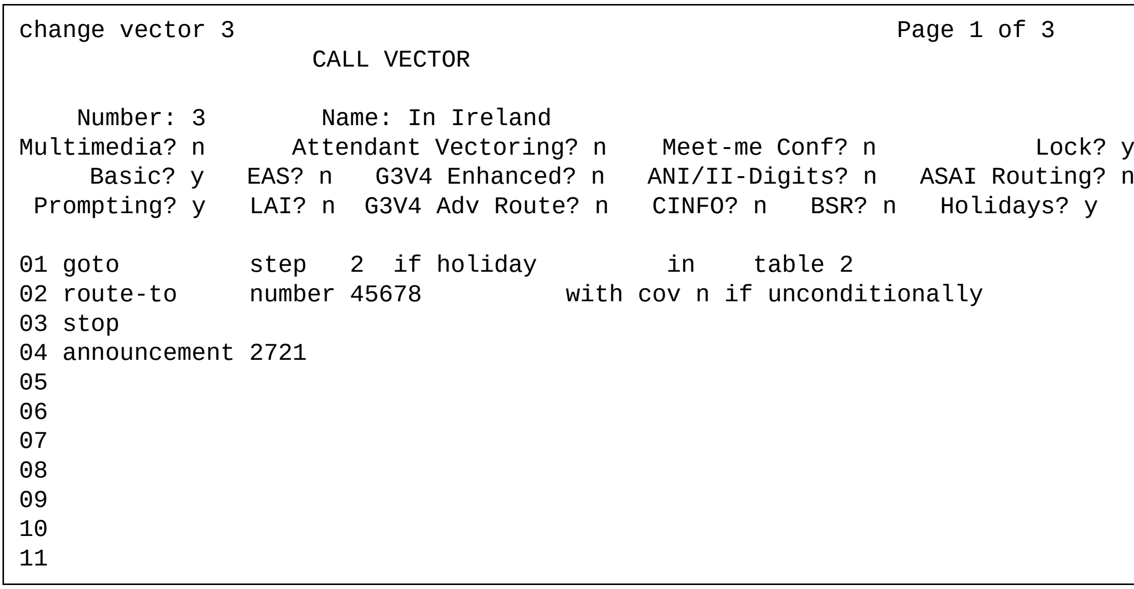
The following examples use **`goto`** commands to route calls for holidays.

Holiday Vectoring 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
```

Holiday Vectoring 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, as shown in the following example, to display which vectors and vector steps are using the selected Holiday Table.

List of Holiday Table use in vectors

```
list usage holiday-table
                             LIST USAGE REPORT

     Used By
     Vector                 Vector Number 1                       Step 1
     Vector                 Vector Number 3                       Step 1
```

Holiday Vectoring considerations

Consider the following when administering Holiday Vectoring:

- Administration of Holiday Tables is supported only on the switch and cannot be changed using adjunct vectoring tools.
- Holiday Vectoring is only available when Vectoring (Basic) or Attendant Vectoring is enabled.
- There is no validation that verifies the consistency among the 15 holidays in any table. If the same holiday is entered twice, the system stops checking with the first entry that is found.
- With holidays that are ranges of dates, the ranges could overlap. When a call is in vector processing, the holidays are checked from top to bottom on the table and the check stops if a match is found. Even though there might be multiple entries that would match, 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, then a the date can be a 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, the holiday tables must be readministered.

- When disabling the Holiday Vectoring feature (changing the value of the `Vectoring (Holidays)` field from y to n on the Customer Options form), the vectors are checked for any **`goto...if holiday`** steps. If any of these steps are found, an error message is displayed, and the change is not allowed. The customer must remove those vector steps first before the feature can be disabled.

Chapter 8: 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.

This chapter includes the following sections:

Command set

The following table lists the commands associated with Meet-me Conference.

Meet-me Conference command set

Command category	Action taken	Command
Information collection		
	Collect information from the calling party.	`collect digits`
Treatment		

	Play an announcement.
	Play a busy tone and stop vector processing.
	Disconnect the call.
	Delay with audible feedback of silence, ringback, system music, or alternate audio or music source.
		`announcement`
		`busy`
		`disconnect`
		`wait-time`

Routing		
	Route to the appropriate meet-me conference and stop vector processing.	`route-to`
Branching/Programming		

	Go to a vector step.
	Go to another vector.
	Stop vector processing.
		`goto step`
		`goto vector`
		`stop`

Information collection commands

Meet-me Conference uses the following information collection commands:

- collect command on page 119

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.

collect command

Syntax: collect 6 digits after announcement <extension>

When the `Meet-me Conf` field is enabled, 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

Attendant Vectoring allows use of several treatment commands, including:

- announcement command on page 119
- busy command on page 119
- disconnect command on page 120
- wait-time command on page 120

The following sections detail the syntax that can be used for these commands and any information that is specific to the Meet-me Conference feature.

announcement command

Syntax: announcement <extension>

The usage for the **`announcement`** command is the same as in Basic Call Vectoring. See the Basic Call Vectoring chapter for details on using this command.

busy command

Syntax: busy

The usage for the **`busy`** command is the same as in Basic Call Vectoring. See the Basic Call Vectoring chapter for details on using this command.

disconnect command

Syntax: disconnect after announcement <extension>

The usage for the **`disconnect`** command is the same as in Basic Call Vectoring. See the Basic Call Vectoring chapter for details on using this command.

wait-time command

Syntax: wait-time <time> secs hearing <silence, ringback, music>

The usage for the **`wait-time`** command is the same as in Basic Call Vectoring. See the Basic Call Vectoring chapter for details on using this command.

Routing commands

Meet-me Conference uses the following routing command:

- route-to meetme command on page 120

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

Meet-me Conference uses several branching/ programming commands, including:

- goto step command *on page 121*
- stop command on page 121

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

Syntax: goto step <step #> if meet-me-idle

Syntax: 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 saying they are the first party to access the call can be given to the caller.

The **`meet-me-full`** condition is used when the Meet-me Conference already has the maximum of six parties on the call.

Syntax: 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

The use of the **`stop`** command is the same as in Basic Call Vectoring. See the Basic Call Vectoring chapter for details on using this command.

Administering Meet-me Conference

This section contains the following information:

Activating the Meet-me Conference feature

Meet-me Conference is available for all switch models that support the R11 call processing software.

To enable the Meet-me Conference feature:

- The G3 Version field of the Customer Options screen must be set to V11 or later.
- The Enhanced Conferencing field of the Customer Options screen must be enabled. This feature has an RTU cost and must be enabled through the License File process.

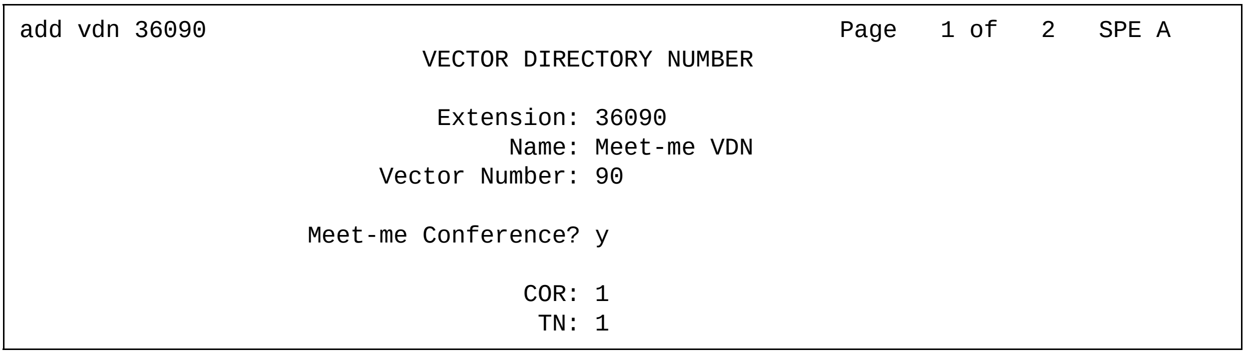
Creating a Meet-me Conference VDN

To create a Meet-me Conference VDN (using example VDN 36090):

1. Enter:

   `add vdn 36090`

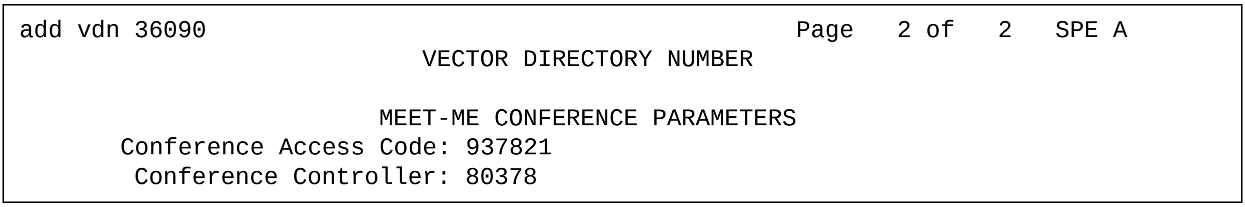
   The system displays the `VECTOR DIRECTORY NUMBER` screen:



```
add vdn 36090                                              Page   1 of   2   SPE A
                            VECTOR DIRECTORY NUMBER

                             Extension: 36090
                                  Name: Meet-me VDN
                         Vector Number: 90

                    Meet-me Conference? y

                                   COR: 1
                                    TN: 1
```

2. Enter a name, a vector number, and enter **y** in the `Meet-me Conference` field.
3. Press **NEXTPAGE** to display page 2.

   The system displays page 2 of the `VECTOR DIRECTORY NUMBER` screen:



```
add vdn 36090                                         Page   2 of   2   SPE A
                            VECTOR DIRECTORY NUMBER

                         MEET-ME CONFERENCE PARAMETERS
       Conference Access Code: 937821
        Conference Controller: 80378
```

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” superuser login.

   SECURITY ALERT:

   You should 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. Press **ENTER** to submit the VDN.

Creating a Meet-me Conference vector

To create a Meet-me Conference vector (using example vector number 90):

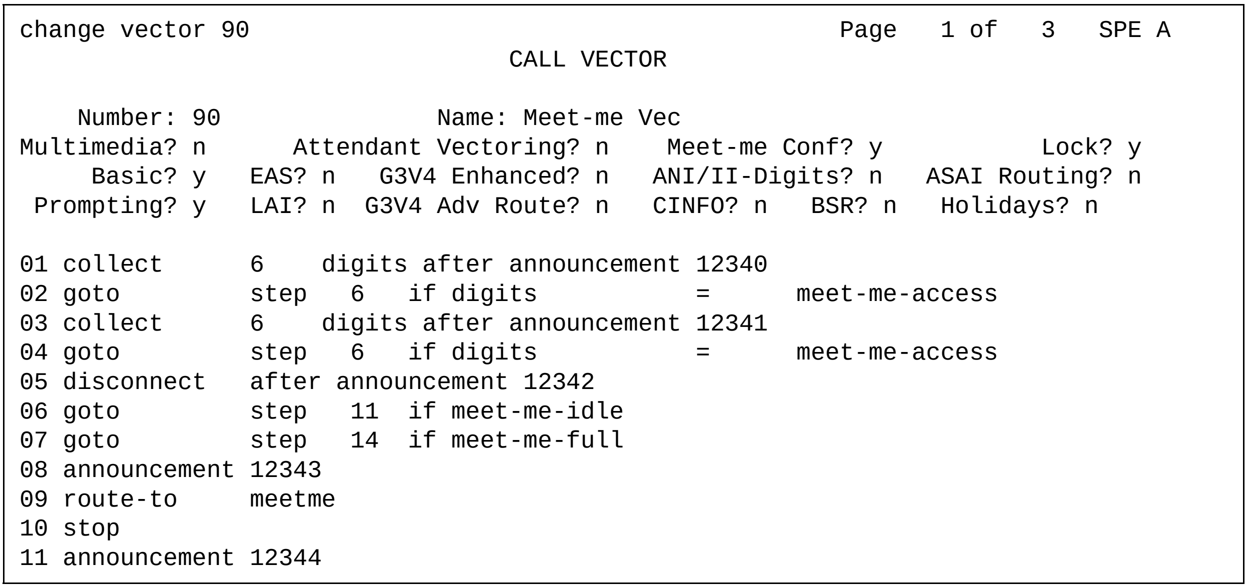
1. Enter:

   `change vector 90`

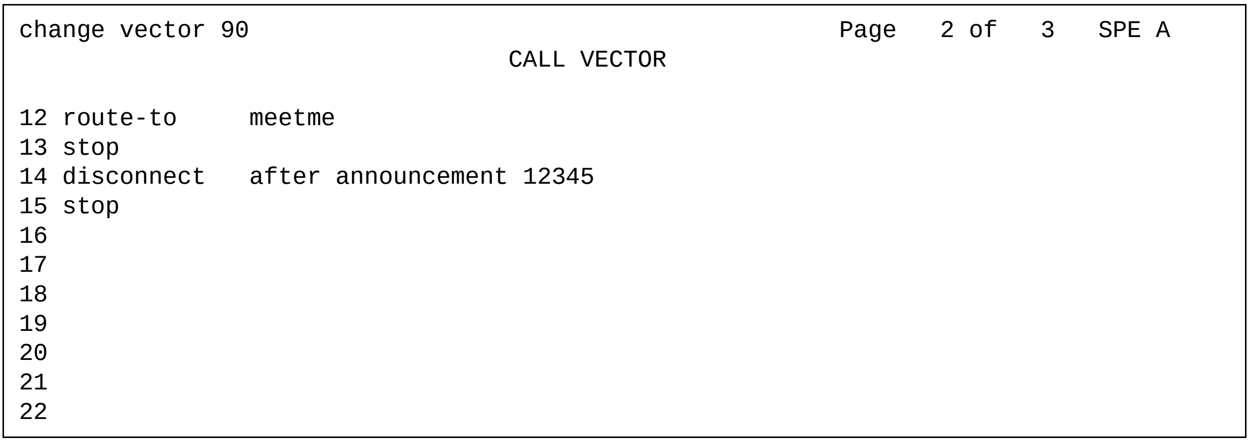
   The system displays the `CALL VECTOR` screen.

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.

Interactions

The following are administration interactions for Meet-me Conference.

General

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, they will be removed from any queue (if applicable) 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. If a record is required, the called party will be the Meet-me Conference VDN number and the duration will be the length of time that the party was included in the call. There will be an individual record for each party that will be output when the party drops from the call. One option that will record 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 the Intra-switch CDR feature is used with the Meet-me Conference VDNs, the condition code should 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 feature is not active for Meet-me Conference VDNs, the creation and contents of call records will depend on the trunk group translations for external callers to the Meet-me Conference. Internal callers to the Meet-me Conference will not generate any records 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, the administrator must first remove all vector steps. To change a non Meet-me Conference vector to a Meet-me Conference vector, the administrator must first remove all vector steps. If either of these conditions exist, a warning message displays that states “VDNs currently assigned to this vector may not operate as expected.” The next time the administrator tries to submit a change to the Meet-me Conference VDN, they would be forced to assign the VDN to a Meet-me Conference vector.

Direct Inward Dialing (DID)

If the VDN extension is part of the customer’s DID block, external users will be able to access the conference VDN. If the VDN extension is not part of the customer’s DID block, only internal callers on the customer’s network or remote access callers can access the conference VDN.

Disabling Enhanced Conferencing

If Meet-me Conference VDNs are assigned when disabling the Enhanced Conferencing option, the change is not allowed and the message “Must first remove all Meet-me Conf VDNs and vectors” is displayed. The administrator must remove those VDNs and vectors before the option can be disabled.

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 message “Must first remove as conference controller on VDN form” is displayed.

Security issues

The Meet-me Conference feature is a potential security problem. If Meet-me Conference VDNs are assigned without access codes, hackers could tie up Meet-me Conference facilities, keeping others from conducting legitimate business, and could potentially access the switch and use the switch to make unauthorized calls. Therefore, we should recommend that all Meet-me Conference VDNs have access codes that are known only to administrators and users on a need to know basis. We should also recommend that access codes be changed on a regular basis to reduce the risk of unauthorized access to the switch.

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. See Tracking unexpected events on page 210.

Capacity issues

Meet-me Conference calls count towards the maximum number of 3-way and 6-way conference calls.

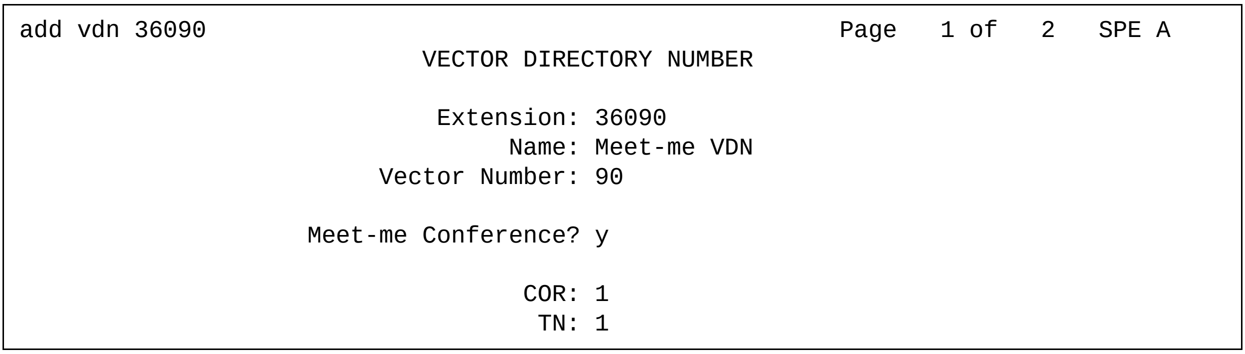
Users cannot add more parties to a conference call once the system maximum is reached.

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 contact center applications is unchanged.

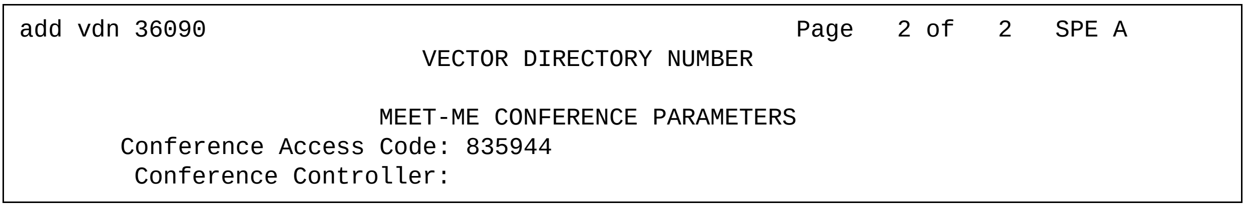
Meet-me Conference call processing scenario

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.



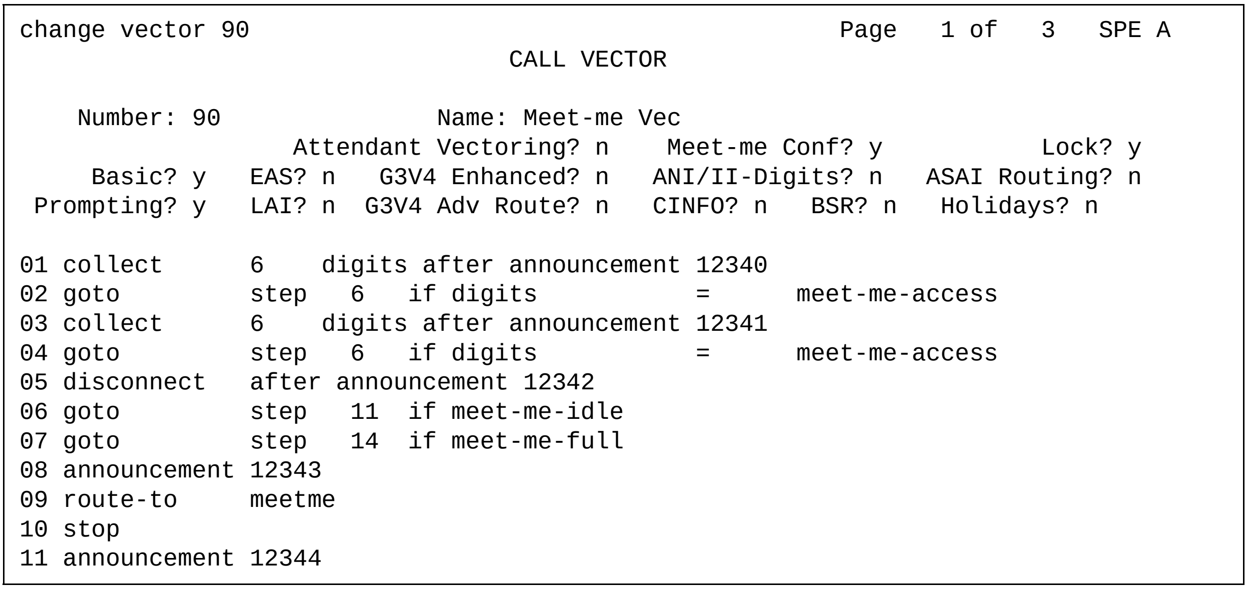
```
add vdn 36090                                            Page   1 of   2   SPE A
                            VECTOR DIRECTORY NUMBER

                             Extension: 36090
                                  Name: Meet-me VDN
                         Vector Number: 90

                    Meet-me Conference? y

                                   COR: 1
                                    TN: 1
```

VDN 36090 is administered with an access code of 835944. See the following screen.

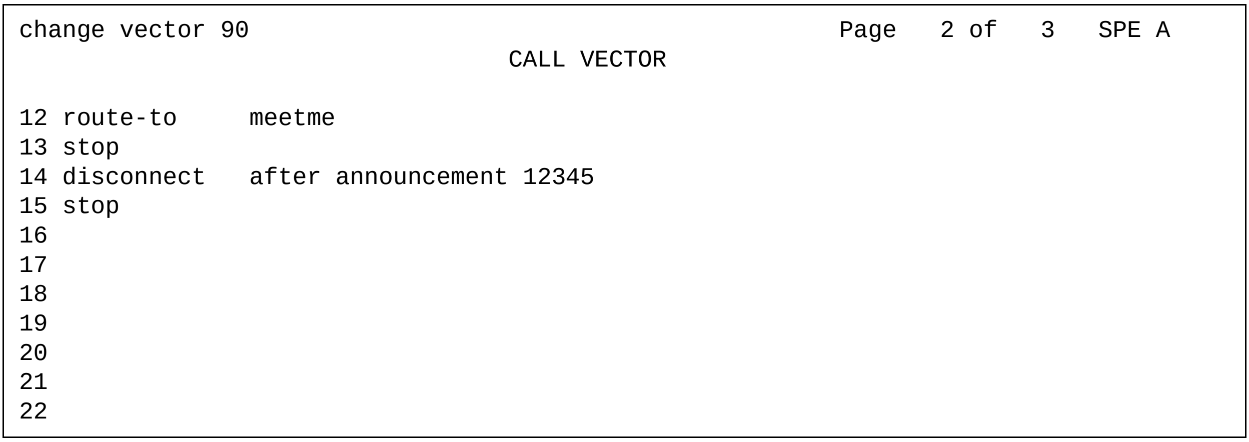


```
add vdn 36090                                         Page   2 of   2   SPE A
                            VECTOR DIRECTORY NUMBER

                         MEET-ME CONFERENCE PARAMETERS
       Conference Access Code: 835944
        Conference Controller:
```

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, the vector processing continues with vector step 6. If the access code is invalid, the 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, the 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. Good-bye.”

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. Good-bye.”

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 will hear an “entry” tone.

When the conference call is over and callers drop out of the conference call, any remaining parties on the call will hear an “exit” tone.

Troubleshooting

This section describes common problems and possible resolutions for the Meet-Me Conference feature. Topics described in this section include:

- Conference call drops on page 129
- Sound volume is too low on page 130

Conference call drops

The conference call drops abruptly for no apparent reason.

Possible reason – The Vector Disconnect Timer on the System-Parameters Features screen is set to a value that does is shorter than the duration of the Meet-Me Conference session.

Solution – Increase the Vector Disconnect Timer value.

Sound volume is too low

Voice volume levels for some conference participants is too low.

Possible reason – The affected conference participants connect through international trunks in which CO loss plans are set for too much loss.

Solution – In the System-Parameters Country Options form, go to Tone & Country Loss Plans (page 3) and change the values specified in the `End-to-End total loss (dB) in a n-party conference` field.

Chapter 9: Call Vectoring applications

This chapter provides example applications of the Call Vectoring feature.

List of example applications

Example applications and the primary feature that is illustrated are listed in the following table.

Example	Features used
Customer service center example on page 132	Basic Call Vectoring
Automated attendant example on page 133	Call Prompting
Data in/voice answer and data/message collection example on page 134	Call Prompting, Basic Call Vectoring
Local attendant group access code on page 139	Attendant Vectoring
Incoming trunk calls to attendant group on page 140	Attendant Vectoring
Incoming LDN calls on page 140	Attendant Vectoring
Night station service example on page 141	Attendant Vectoring
Holiday Vectoring example on page 142	Holiday Vectoring
Dial by Name on page 144	Basic Call Vectoring, Call Prompting

Customer service center example

The example scenario involves a customer service center that is open weekdays from 8 a.m. until 5 p.m. The center provides two separate telephone numbers. One number is for regular customers, while the other number is for 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 if time-of-day is all 17:00 to all 08:00
      2. goto vector 29 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 if time-of-day is all 17:00 to all 08:00
      2. goto vector 29 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 to the correct number, vector 22 is accessed. The first two steps of this vector determine if the call arrives during nonbusiness 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 vector 29 is accessed, the caller is given the appropriate announcement twice (skills 1 and 3) and is then disconnected (step 3).

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 the queue, control is sent to step 12, which routes the call to the attendant. If less than 10 calls are in the due, the call is queued to the main split (step 4). If the call is not answered immediately, an appropriate announcement is provided (step 5), followed by a wait period (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 spilts, 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.

A call that is placed by a nonpriority customer is processed by vector 21. Vector 21 provides a treatment similar to that provided by vector 22, with the following exceptions:

- Backup splits are not queried for nonpriority calls
- Priority calls are assigned a higher priority in the queue
- Priority calls route to an operator when too many calls are queued, but nonpriority calls route to a busy signal.

Automated attendant example

This example scenario shows the use of Automated Attendant, which is one of the applications that can be supported by the Call Prompting feature. Automated Attendant allows the caller to enter the extension of the party that the caller wants to reach. Depending on the parameters established, the user can enter up to 16 digits from a touchtone telephone.

Automated Attendant is usually used by contact centers that do not have DID trunks and whose callers know the extension of the people they are calling. Because it reduces the need for “live attendants,” Automated Attendant reduces contact center costs.

The following example shows an example of a vector that implements Automated Attendant.

Example application - automated attendant

```
1. wait-time 0 seconds hearing ringback
     2. collect 5 digits after announcement 30001
        (‘‘You have reached Ridel Publications in Greenbrook.
        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
```

Step 1 of this 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 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 he or she wants to reach (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 because the digits entered do not comprise a valid extension, then the **`route-to number`** command in step 4 routes the call to the attendant. However, as long as the destination is a valid extension, the **`route-to digits`** command succeeds, coverage applies, and vector processing terminates. Note that even if the destination is busy, vector processing terminates because coverage call processing takes effect.

Data in/voice answer and data/message collection example

This example 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 also uses Call Prompting to determine the desired service. Three services are offered:

- New accounts enables the customer to open a new account.
- Account inquiries enables the customer to make inquiries concerning his or her account.
- Net asset values enables the customer to hear information concerning the net asset values of the company’s funds.

If the caller selects “account inquiries,” he or she is prompted to input his or her account number before being answered by an agent. The agent can use the CALLR-INFO button to display this 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 CallMaster VI telephone is not supported on Category B systems.

This example uses three other applications that can be supported by the Call Prompting feature:

- Data In/Voice Answer (DIVA) allows a caller to receive information on a topic that he selects at the prompt. The caller selects the desired topic by entering the appropriate digits.
- Data Collection provides a method of collecting digits from a caller. The requested digits comprise an official number of some sort. For example, a Social Security Number, and they help the system process the call more efficiently.
- Message Collection allows the caller to leave a recorded message instead of waiting for the call to be answered.

The four vectors shown below illustrate how the mutual fund company handles telephone calls. Typically, the vector should be 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’re sorry.  All of our operators are busy at
         the moment.  If you’d like to leave your name and
         telephone number so that we can get back to you,
         dial 1.’’)
      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’re sorry, we cannot take
         your message at this time.  You may continue to hold, or
         you can call back later.’’)
     12. goto step 4 if unconditionally
```

DIVA and data/message collection vector examples (continued)

```
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’re sorry. All of our operators are busy at
         the moment. If you’d like to leave your name and
         telephone number so that we can get back to you,
         dial 1.”)
      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’re sorry, we cannot take
         your message at this time. You may continue to hold, or
         you can 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 upon the service desired. If the caller enters a digit other than 1, 2, or 3 mentioned, or if the caller fails to enter any digits within 10 seconds, then the command fails and the call is routed to the attendant (step 6). If the caller enters 1, 2, or 3 within 10 seconds, then the call is routed to the vector specified in the appropriate **`route-to number`** command, which appears in steps 3, 4, and 5.

For instance, assume that, when prompted, the caller enters 3 because he or she wants to learn about the net asset values of the company’s funds. In such a case, the **`route-to number`** commands 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 for 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 net asset values and then disconnects the call.

The process just described, whereby the caller receives information as a result of making a request at the prompt, is an example of the Data In/Voice Answer (DIVA) application.

Returning to the main vector, suppose that another caller wants to make an inquiry into his or her account, and the caller enters 2 when prompted. In such a case, 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 his or her 6-digit account number. The command then collects the digits that are entered by the caller. Whether or not the caller correctly enters the digits, the **queue-to split** command in step 4 queues the call. If an agent does not immediately answer the call, the standard announcement is provided in step 5 and, if necessary, a delay is provided in step 6. The announcement in step 7 provides the caller with the option of leaving a message instead of having his or her call wait in queue. The caller is instructed to enter 1 if he or she wants 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 Message Center split so that the caller can leave a message. If the connection is made, the caller first 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 could not be made. Thereafter, the **goto step** command in step 14 sends call control back to step 6, which leads the caller back into the steps to leave a message.

The process that was just described, whereby the caller, when prompted, enters digits that comprise an official number (an account number, in this case), 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 that are entered by the caller. As a result, the agent need not request the account number from the caller.

Finally, suppose that a third caller wants to open an account and that he or she enters 1 when prompted in the main vector. In this case, 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, if necessary, step 3 provides the appropriate announcement, and step 4 provides a delay period. The announcement in step 5 provides the caller with the option of leaving a recorded message instead of having his or her call wait in queue. This is an example of the Message Collection application. The caller is instructed to enter 1 if he or she wants 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 so that the caller can leave a message. If the connection is made, the caller first 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 could not be 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.

Attendant routing example

The following example show how the Attendant Vectoring commands can be used to route calls in an attendant environment. For the attendant vectors, consider the following vectors and vector administration.

Note:

For the following vector examples, tenant partitioning is turned on:

Attendant Vectoring vectors

VDN 1999 **vector 1**	**VDN 2999** **vector 2**	**VDN 3999** **vector 3**
1. wait-time 0 secs hearing ringback 2. goto step 6 if time-of-day is all 12:00 to 13:00 3. queue-to attd-group 4. goto step 7 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 6 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 step10 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 7 if time-of-day is all 12:00 to 13:00 3. queue-to attd-group 4. goto step 7 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 10 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 VDNs 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.

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 terminated to the attendant and vector processing terminates. Otherwise, the call is queued to the attendant group and the caller hears music from the 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, in the example above, if it is lunch time the call will route to the hunt group and all hunt group processing will then apply. 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 call is received on a trunk that has the attendant group assigned as the incoming destination or when the call is addressed to the attendant group, 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 queueing 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 cannot be queued, the call is routed to the remote location and vector processing terminates.

Note:

The main difference from the example shown in Local attendant group access code on page 139 is queueing the call to the hunt group rather than routing the call there. In this example, the call will not follow the hunt group's coverage path or night service destination.

Incoming LDN calls

When a call is received for an 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 into 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 attendant 6000's queue, 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 terminated 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 TN 3 every 15 seconds. If the call cannot be queued, it is sent to attendant 6000.

Note:

Vector 3 attempts to queue the call to attendant 6000. A **`route-to`** command could also be used, but care should be taken since an attendant cannot be assigned a coverage path.

Night station service example

This example shows how you can use the Attendant Vectoring features 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 example

This example is a vector that is directing calls to special processing because of a holiday or special event. Holiday Vectoring is an enhancement that simplifies vector writing for holidays. It is designed for customers who need to reroute or provide special handling for date-related calls on a regular basis.

In this example, a commercial bank that is headquartered in Germany has branches in Europe. The bank recently established a U.S. presence by opening branches in the New York City metropolitan area. The bank's credit card division operates two 100-agent contact centers in Ireland and Germany and one 50-agent contact center in the U.S.

All agents in the European centers are bilingual and assigned to splits that handle calls from both English and German customers. The same is true for the agents in the New York contact center. Because the New York contact center is open 24 hours a day, it often takes calls that are routed from the Irish and German contact centers after those centers close at 6:00 p.m. local time.

Due to the large number of bank holidays per year in Europe (up to 30 days), the Holiday Vectoring feature can be used to create vectors that distribute calls automatically on holidays. The contact center administrator recommended this feature to the systems administrator to save the cost of time spent on writing vectors for date-related processing, and to save business that would be lost to abandoned calls if vectors are not readministered for holidays.

The following figure indicates that, beginning on December 24 and continuing through 6:00 am on January 2, incoming calls to the contact 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 had to 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              12    31               Christmas
    01    01  00   00     01    02  06   00      New Year’s Day
```

After submitting the Holiday Tables form, the next step is to modify the vector processing for these holidays. On the Call Vector form, enter the new goto conditional for the holidays.

Modifying a vector to route according to a holiday table

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

    Number: 3        Name: In Ireland
Multimedia? n     Attendant Vectoring? 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
```

The setup for the vector routes the call to the United States contact center. For example, if someone in Europe calls the bank before 6:00 a.m. on January 2, the call is routed to the United States contact center. If someone in Europe calls after 6:00 a.m. on January 2, the call is routed to the German contact center.

Dial by Name

The Dial by Name feature allows you to “dial” someone by entering the person’s name from your touch-tone keypad. This feature is accessible by using the Call Vectoring feature and the integrated announcement circuit pack to create an “auto-attendant” procedure in which one of the options allows callers to enter a person’s name instead of the person’s extension number. The system processes the name characters received, and, when a match is found, the number is dialed automatically.

Note:

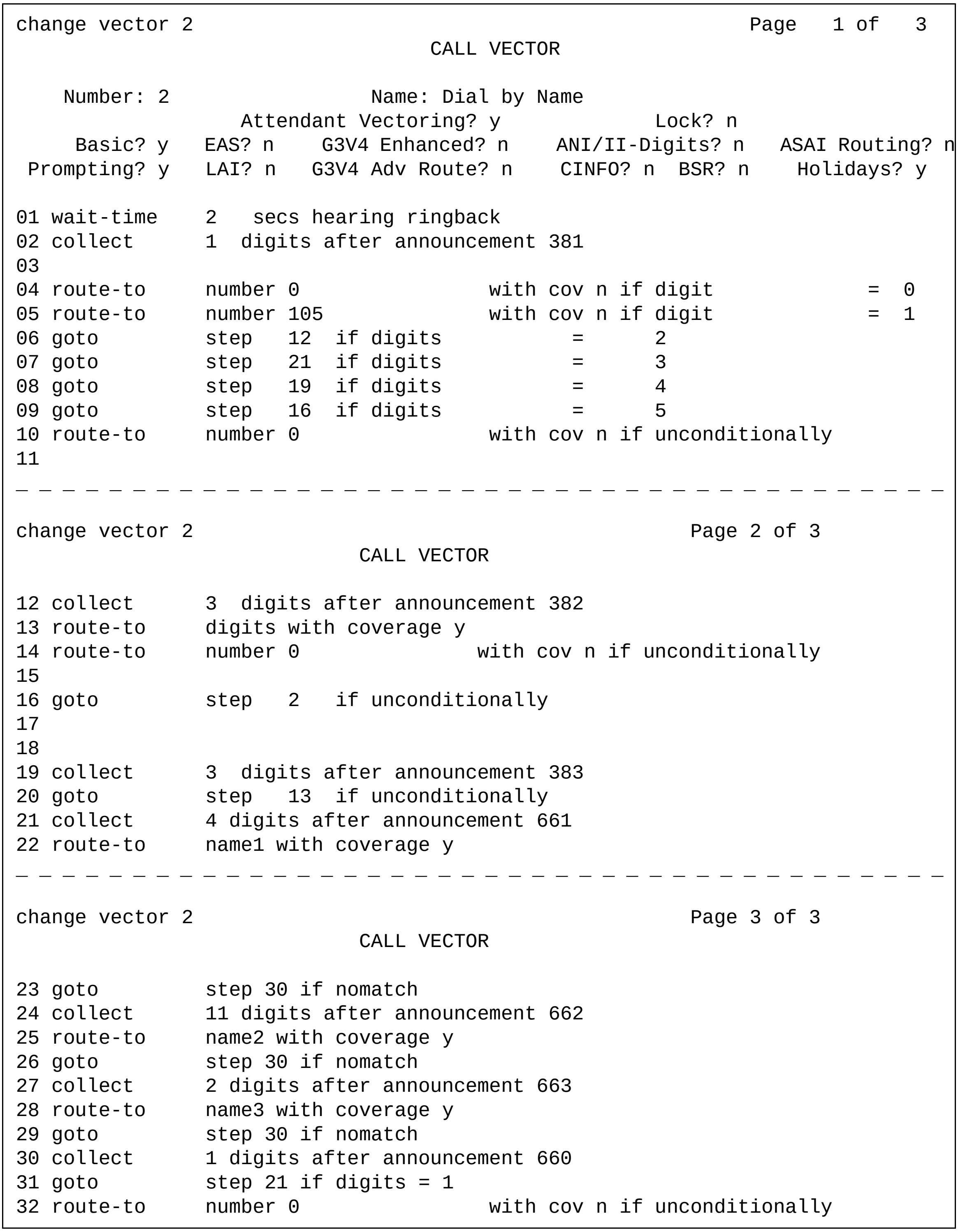
> The Dial by Name feature must be enabled to create a vector for this purpose.

A typical scenario includes the following call processing features:

- When a call comes in to the system (usually to a Listed Directory Number), a vector routes the call to an announcement that says, “Hello. You have reached A1 Hotel. Please press 0 for the operator; press 1 for the front desk; press 2 if you know the guest’s extension; press 3 if you know the guest’s name; press 4 if you want to choose from a list of extensions; or press 5 if you wish to hear these options again.”
- When the caller selects 3, the caller is then instructed to enter the person’s name.
- As soon as a single match is found, the call is placed to that person.

You can assign several vectors that define how calls will be handled 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 ringback for 2 seconds.
2. Announcement 381 plays. This announcement asks the caller to do one of the following:
   - Press 0 if the caller wants the operator; if the caller presses 0 or waits for the timeout, the call is routed to the operator.
   - Press 1 if the caller wants the front desk; if the caller presses 1, the call is routed to extension 105, which is the front desk.
   - Press 1 if the caller knows the person’s extension; if the caller presses 2, the call is routed to announcement 382, which instructs the caller to dial the person’s extension.
   - Press 3 if the caller knows the person’s name; if the caller presses 3, the following sub-procedure occurs:
     i. Announcement 661 plays requesting that the caller enter the first four characters of the person’s last name.
        — If there is a single match, the call is redirected.
        — If there are multiple matches, continue with ii.
        — If there is no match, go to iv.

     ii. Announcement 662 plays requesting that the caller enter the rest of the person’s last name, followed by the # key.
        — If there is a single match, the call is redirected.
        — If there are multiple matches, continue with iii.
        — If there is no match, go to iv.

     iii. Announcement 663 plays requesting that the caller enter the first two characters of the person’s first name.
        — If there is a single match, the call is redirected.
        — If there is no match, continue with iv.

     iv. Since there are still no matches, announcement 660 plays telling the caller that he or she can press 1 to try again, or press 0 to get an operator.
   - Press 4 if the caller knows the department (such as housekeeping) that he or she 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 over again; if the caller presses 5, the caller hears announcement 381, which repeats all of the options.
   - If the caller dials anything else, the call is routed to the operator.

Vectors exercises

This section presents several typical business scenarios that involve telephone use. One or more vectors are provided that show how to handle each of these scenarios.

The vectors presented here are intended to be “suggested solutions.” Individual contact centers must consider their own unique requirements and budget in selecting and writing vectors.

Emergency and routine service

Write a vector that does the following:

- Delivers the following message to handle emergency calls: “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.”
- Enables the caller to speak with an agent, if an agent is available, concerning a nonemergency matter.

Suggested solution 1

Call Vectoring option

```
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’re sorry.  All of
     our operators are busy.  Please hold.")
8.   wait-time 10 seconds hearing music
9.   goto step 7 if unconditionally
10.  disconnect after announcement 4200 ("We’re sorry.
     All of our operators are busy at the moment. Please call back at
     your convenience.")
```

In step 2 of the example vector shown above, the **`announcement`** command provides the caller with the appropriate emergency information, and it invites the caller to hold if he or she wants to speak with an operator on another matter. If the caller holds, the caller hears several seconds of ringback provided by the **`wait-time`** command in step 3. Thereafter, the **`goto step`** command in step 4 checks whether 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 first 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 (if necessary) that is provided by 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.

Suggested solution 2

Note:

This example uses the Call Prompting feature. For more information about Call Prompting, see Chapter 5: Call Prompting on page 73.

Call Vectoring and Call Prompting option

```
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.
         Otherwise, please hang up now.")
      3. route-to number 1031 with cov y if digit = 1
      4. announcement 3100 ("Entry not understood.  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’re sorry.  All of
         our operators are busy.  Please hold.")
      7. wait-time 10 seconds hearing music
      8. goto step 6 if unconditionally
      9. disconnect after announcement 4200 ("We’re
         sorry.  All of our operators are busy at the moment.
         Please call back at your convenience.")
```

Suggested Solution 2 involves both Call Vectoring and Call Prompting. Also, it involves two vectors instead of just one vector, and it assumes the that caller is calling from a touchtone telephone. The announcement portion of the **`collect digits after announcement`** command in step 2 of Vector 30 first provides the caller with the appropriate emergency information. It then invites the caller to press “1” if the caller is calling for some other reason. If this is not the case, it finally suggests that the caller hang up.

Assume that the caller wants to hold the line but enters the incorrect touchtone digit (2, for example). In such a case, 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 was entered, the call is not routed to the VDN extension. Instead, control is passed to step 4, where the **`announcement`** command first informs the caller of the input error and then invites 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 Caller Treatment

The contact center is staffed by union agents who work under a contract that stipulates that agents are free to leave promptly at 5:00 p.m. However, you are concerned about the callers who will call shortly before 5:00 p.m. on any given day and find themselves waiting in queue the at the top of the hour.

Write a vector that warns late callers that their call may not be serviced. Remember that business hours are from 8:00 a.m. to 5:00 p.m., Monday through Friday.

Suggested solution:

Late caller 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 ("It is almost closing time.
     We will try to service you before we close for the day.
     However, if we are unable to do so, please call back
     at your convenience 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’re sorry, our office is now closed.
     Please call back at your convenience between 8:00 A.M.     and 5:00 P.M.,
     Monday through Friday.")
15.  disconnect after announcement 400 ("We’re sorry, our office is     closed.
     Please call back at your convenience between 8:00 A.M.     and 5:00 P.M.,
     Monday through Friday.")
16.  disconnect after announcement 500 ("We’re sorry, we cannot service your
     call at this time. Please call back at your convenience between
     8:00 A.M. and 5:00 P.M., Monday through Friday.")
```

In the example vector shown above, specific treatment is provided for calls that come into the switch after working hours, during the weekend, or as the working day comes to a close.

The **`goto step`** command in step 1 checks whether 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 whether 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 whether the current time is any time 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 the 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 the appropriate late caller announcement is provided to the caller. 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 whether 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 at least a second after 5:10 P.M., control is passed to step 14 as described previously.

Messaging option

Write a vector that:

- Does the following if the oldest call waiting is in queue for longer than 75 seconds:
  - — Sends the call to AUDIX (if possible)
  - — Delivers to the caller the following personalized AUDIX message: “All of our MegaSports agents are busy...Please leave your name and telephone number.”
- Plays 30 seconds of ringback for the caller
- After the ringback, plays an announcement for the caller that is followed by music

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're sorry, all of our
     MegaSports agents are busy.  If you'd like to leave a
     message, please do so 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're 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 shown above checks whether the oldest call waiting in split 50 has been waiting for 75 seconds or more. If so, control is passed to step 8, where the **`announcement`** command first informs the caller that all of the agents are busy and then invites the caller to either call back at the appropriate time or leave a recorded 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 that he or she 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 that provides the caller with the appropriate apology and subsequent directives. 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 for 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.

Appendix A: Call Vectoring commands

This appendix provides information about the commands used in Call Vectoring, including:

- A table that lists the Comunication Manager options that are required to enable various vector commands
- A table that contains a brief description of each command's function and also the appendix page where the command can be referenced
- A job aid tables that graphically illustrate how to use the Call Vectoring commands
- A manual page directory that details the purpose and function of the Call Vectoring commands and also any relevant interactions involving the commands

Comunication Manager options required to enable vector commands

The following table lists the options that are required to enable various vector commands, options, and parameters.

Vector commands / required options summary

Command	Basic	Prompting	Attendant	Other Options Required
`announcement`	x	x		
`busy`	x			
`check best`	x			ACD; G3V4 Advanced Routing; Best Service Routing
`check split if <condition>`	x			ACD

Vector commands / required options summary (Continued)

Command	Basic	Prompting	Attendant	Other Options Required
check split if wait-improved	x			ACD; G3V4 Advanced Routing; Best Service Routing
collect digits		x		
disconnect	x		x	
disconnect after announcement <extension>	x		x	
goto step/vector if unconditionally	x	x		
goto step/vector if digits		x		
goto step/vector if oldest-call-wait pri	x			ACD; G3V4 Enhanced
goto step/vector if queue fail			x	
goto step/vector if holiday in/not-in table	x		x	Holiday Vectoring
messaging split	x	x		
queue-to split	x			ACD
queue-to attd-group				Attendant Vectoring
queue-to attendant				Attendant Vectoring
queue-to hunt group				Attendant Vectoring
route-to number	x			
route-to digits with cov y (n)		x		
route-to number if digit		x		
route-to number if unconditionally with cov y (n)[1]	x	x		
route-to number if digit with cov y (n)[1]		x		

Vector commands / required options summary (Continued)

Command	Basic	Prompting	Attendant	Other Options Required
`route-to number if unconditionally`	x	x		
`stop`	x	x		
`wait-time <time>`	x	x	x	
`wait-time <time> hearing <treatment>`	x	x	x	

1. If G3V4 software has not been purchased, these commands require the G3V4 maintenance load.

Command description/reference

The following table provides a brief description of the function of each of the Call Vectoring command. See the listed page number for a complete description of the command.

Command description/reference table

Command	Function
Announcement command on page 161	To connect caller to delay recording.
Busy command on page 163	To connect caller to busy tone.
Check command on page 164	To connect/queue call on a conditional basis.
Collect Digits command on page 166	To prompt caller for digits.
Disconnect command on page 170	To force disconnect of call with optional announcement.
Goto step and goto vector commands on page 172	To cause unconditional/conditional branch to another step in the vector.
Messaging command on page 175	To allow caller to leave message for callback.

Command description/reference table (Continued)

Command	Function
Queue-to command on page 177	To connect/queue call to the primary split/skill or to the best resource (or, with Attendant Vectoring, attendant, attendant group, or hunt-group) found by a consider series.
Route-to command on page 180	To connect call to destination entered via **`collect digits`** command, or to connect call to internal/external destination.
Stop command on page 185	To stop further vector processing.
Wait-time command on page 186	To initiate feedback to caller (if needed) and delay processing of the next step.

Command job aid

The vector command job aid shown below lists the Call Vectoring commands, together with the various conditions, and parameter options and values that are available for use with each command.

Obtaining switch capacity information

Most vector commands require one or more input values for the command, as well as for various parameters, such as an announcement extension number, a time interval, a maximum queue size, and so forth. When the minimum and maximum ranges for command parameter values are identical for all Avaya switch platforms, the limiting ranges are specified in the job aid. Alternately, when the minimum and maximum ranges for a parameter value are not the same among the among Avaya switch platforms, the upper limit of a value range is indicated by the term “switch max.”

To determine the maximum values you can use in Call Vectoring commands, see *System Capacities Table for Avaya Communication Manager on Avaya Media Servers*, 555-233-605. To access the document online, go to:

http://avayadocs.com

Vector command job aid for GuestWorks and BCS

Command	Parameters / Conditions			
announcement ________ extension no.				
busy				
check				
split _____ 1 to switch max	**pri** ___ **l**ow-priority **m**edium-priority **h**igh-priority **t**op-priority	**if** ________ available-agents calls-queued oldest-call-wait[2] staffed-agents unconditionally	___ **>** **<** **<=>** **>**	___________ 0 to 1499[1] 1 to 999[1] 1 to 999 seconds 0 to 1499[1]
collect ____ **digits**				
1-16 digits	**after announcement** _____ extension no. or “none”			
disconnect	**after announcement** _____ extension no. or “none”			
goto				
step _____ 1-32 or **vector** ______ 1 to switch max				
if				
available-agents				
in split _____ 1 to switch max		____ <,= >,=	____ 1 to 1500[1] 0 to 1499[1]	
calls-queued				
in split ____ 1 to switch max	**pri** ___ **l**ow-priority **m**edium-priority **h**igh-priority **t**op-priority	____ <,= >,=	____ 1 to 999[1] 0 to 998[1]	
digits ___	______			
<,>,=	1-16 digits, including 0-9, “#”, “?”, or “none”[3]			
=	meet-me-access[4]			

Vector command job aid for GuestWorks and BCS (Continued)

Command	Parameters / Conditions			
goto step (or vector) (*continued*)				
if (*continued*)				
holiday ____ in, not-in	**table** ____ 1-10			
meet-me-full[4] **(go-to step, only)**				
meet-me-idle[4] **(go-to step, only)**				
no match[5]				
oldest-call-wait	**in split** ____ 1 to switch max	**pri** ___ **l**ow-priority **m**edium-priority **h**igh-priority **t**op-priority	____ <,>,=	____ 0 to 999 seconds
queue-fail[6]				
staffed-agents	**in split** ______ 1 to switch max		____ <,>,=	____ 1 to 1500[1]
unconditionally				
messaging				
split ____ 1-switch max		**for extension** ____ extension no., “latest” or “active”[7]		
queue-to				
attd-group[6]				
attendant[6]	_________ extension no.			
hunt-group[6] _____ 1 to switch max		**pri** ___ **l**ow-priority **m**edium-priority **h**igh-priority **t**op-priority		
split ____ 1 to switch max		**pri** ___ **l**ow-priority **m**edium-priority **h**igh-priority **t**op-priority		

Vector command job aid for GuestWorks and BCS (Continued)

Command	Parameters / Conditions			
route-to				
	digits	**with coverage** _____ **y**es or **n**o		
	meetme[4]			
	number ____ 0-9, * , # , ~p, ~m, ~s, ~w, ~W, or a leading ~r[8]	**with cov** ___ **y**es or **n**o	**if** ____ digit unconditionally	
	name1 [5]	**with coverage** _____ **y**es or **n**o		
	name2 [5]	**with coverage** _____ **y**es or **n**o		
	name3 [5]	**with coverage** _____ **y**es or **n**o		
stop				
wait-time				
	____	____	**hearing**	______________
	0-999	secs		music, ringback, silence
	0-480	mins		
	0-8	hrs		

1. Maximum limit less on some platforms. Use the help key for your switch administration software to determine the applicable limit for your system.
2. When **oldest-call-wait option** is selected, **pri** level must be set to **low**.
3. The question mark (**?**) is a wild card that matches any digit (0-9) at the specified position. The plus sign (**+**) matches any or no characters at the specified position
4. This item is available in **meet-me conference** vectors, only.
5. This item available with Dial by Name feature, only.
6. This item available with Attendant Vectoring feature, only.
7. "**Active**" refers to the VDN specified by VDN Override settings. "**Latest**" refers to the VDN specified for the current vector.
8. When the specified number is preceded by **~r**, Network Call Redirection is attempted.

Command directory

The manual page directory in this section lists and discusses all of the Call Vectoring commands:

- Announcement command on page 161
- Busy command on page 163
- Check command on page 164
- Collect Digits command on page 166
- Disconnect command on page 170
- Goto step and goto vector commands on page 172
- Queue-to command on page 177
- Route-to command on page 180
- Stop command on page 185
- Wait-time command on page 186

Announcement command

Purpose

Provides the caller with a recorded announcement.

Syntax and valid entries

```
announcement______
        valid extension no.
```

Requirements

Integrated board, aux trunk or analog (T&R or Lineside DS1) announcement equipment must be installed.

Appropriate announcements need to be administered and recorded. For more information, see “Managing Announcements” in *Administrator Guide for Avaya Communication Manager*, 555-233-506.

Operation

The announcement is played from beginning to end unless an agent becomes available. In such a case, the announcement is interrupted and (if manual answering operation is assigned to the agent, or if calls are delivered to the agent on a manual answering basis) ringback is provided. If the call is queued, the call remains as such while the announcement is played. Any feedback that is provided before an announcement (for example, a wait with music or ringback) continues until the announcement is played.

If the announcement’s queue is full, the call retries the announcement step for an indefinite period of time before any new vector steps are processed.

The **`announcement`** command step is skipped, and vector processing continues at the next vector step, whenever any of the following conditions exist:

- Requested announcement is busied out, not available, or not administered.
- Integrated board is not installed.
- External aux trunk or analog equipment is not attached.

For a complete description of the types and operation of announcements see “Managing Announcements” in *Administrator Guide for Avaya Communication Manager*, 555-233-506.

Answer supervision considerations

Unless answer supervision has already been sent, it is sent as soon as the command starts to process the call (even before the announcement starts).

Feature interactions

The command is considered a call acceptance vector command whenever one of the following is true:

- Announcement is available.
- Call is queued for an announcement.
- Announcement is retried.

The command is considered a neutral vector command whenever the announcement is unavailable.

BCMS interactions

The command is not tracked by BCMS.

Busy command

Purpose

The **busy** command gives the caller a busy signal and causes termination of vector processing.

Syntax

```
busy
```

Requirements

No special requirements.

Operation

The command takes effect on non-CO trunk calls whether or not answer supervision has been sent. However, if the call is on a CO trunk and answer supervision has not been sent, the busy is not passed back by the CO, and the caller continues to hear ringback from the CO. Calls are dropped approximately 45 seconds after the busy tone is applied.

If ISDN-PRI is involved, the application of the busy tone is enabled via D-channel messaging. The network switching office returns the busy tone to the caller. The facility to the switch is dropped, thus making it immediately available for another call.

Answer Supervision Considerations

After the 45 second timeout, an unanswered CO trunk call is answered and then dropped. All other unanswered calls after this timeout are dropped without being answered. For an ISDN call that has not yet queued or been answered, no timeout occurs, and answer supervision is not sent. Instead, a message requesting a busy tone is sent to the network and, subsequently, the trunk is released.

Feature interactions

None.

BCMS interactions

A call that is forced busy due to the command is tracked as OTHER in the VDN Report.

Check command

Purpose

Checks the status of a split for possible termination of the call to that split.

Syntax and valid entries

Command	Parameters / Conditions			
check				
split _____	**pri** ___	**if** ________	___	___________
1 to switch max	**l**ow-priority	available-agents	**>**	0 to 1499[1]
	medium-priority	calls-queued	**<**	1 to 999[1]
	high-priority	oldest-call-wait	**<=>**	1 to 999 seconds
	top-priority	staffed-agents	**>**	0 to 1499[1]
		unconditionally		

[1] Maximum limit less on some platforms. Use the help key for your switch administration software to determine the applicable limit for your system.

Requirements

No special requirements.

Operation

The **check** command checks the status of a split against conditions specified in the command. If the conditions specified in the command are met, the call is terminated to the split. If the conditions are met but no agents are available, the call is queued to the split and waits for an agent to become available.

Each **check** command may be used with the keyword **split**. The **check split** command requires you to specify the split to be checked.

The command is customized to check for and/or respond to specific conditions. For example, the command can queue/terminate unconditionally. The command can also queue/terminate if any of the following is true:

- Number of available agents is greater than the threshold value.
- Number of staffed agents is greater than the threshold value.
- Number of calls queued for a specified priority level or higher is less than the threshold value.

- Oldest call waiting in queue at the specified priority level or higher has been waiting less than the threshold value, which is expressed in seconds.

A call may be queued to up to three splits simultaneously. A call remains queued either until vector processing terminates (via a successful **`disconnect`**, **`busy`**, or **`route-to`** command, or via an abandoned call), the call is routed to another VDN (by a **`route-to number`** or **`route-to digits`** command), or the call reaches an agent. When an agent becomes available in any split to which the call is queued, the following actions take place:

- Call begins ringing the agent.
- Call is removed from any other queues.
- Vector processing terminates.

If the desired backup split is one of the splits to which the call is already queued, the call is requeued at the new priority level, provided that the command conditions are met. The step is skipped, and vector processing continues at the next step if any of the following conditions are true:

- Command conditions are not met.
- Desired split's queue is full.
- Desired split has no queue and also no available agents.
- Desired split is not vector-controlled.
- Call is already queued to this split/skill at the specified priority level.
- Call has been previously queued to three different splits.

Note:

A **`route-to`** to another VDN can be used to remove the call from the splits it is queued to if necessary. The steps in the routed-to vector then can be used to queue to other splits.

Answer Supervision Considerations

No answer supervision is returned.

Feature interactions

The **`check`** command can access an AUDIX/Message Center/Server split in cases where a VDN is assigned as a coverage point. To enable this function, the split must be assigned as a vector-controlled hunt group.

The command is considered a call acceptance vector command whenever one of the following is true:

- Call terminates to an agent.
- Call queues to a split.

The command is considered a neutral vector command when the call neither terminates nor queues.

No COR checking is carried out when a **check** step places a call to a split.

The **oldest-call-waiting** condition can check only priority level l (low).

BCMS interactions

The total number of calls to the VDN that are queued via the command and then answered by an agent within a specified time period is tracked as ACD Calls in the VDN Report. The average time that calls spend in a vector before being connected via the command as an ACD call to an agent is tracked as AVG SPEED ANS in the same report.

Collect Digits command

Purpose

The **collect digits** command allows the user to enter up to 16 digits from a touch-tone phone or an internal rotary phone.

Syntax and valid entries

Command	Parameters / Conditions
collect ____ digits	**after announcement _____** extension no. or “none”
1-16 digits	
ced or cdpd	

Requirements

At least one TN744 Call Classifier circuit pack or TN2182 Tone Clock circuit pack must be in the system unless the command is used only to collect digits returned by a VRU or sent by the network and never to collect digits from a caller.

Operation

The collect command has one mode of operation:

- Collecting digits on the switch

Collecting Digits on the switch – The `collect digits` command allows a caller to enter digits from a touch-tone or an internal rotary phone. An optional announcement may be used to request the caller to enter these digits. The announcement can instruct the user to enter an asterisk (*) if incorrect data is entered. When the caller enters an asterisk, the digits collected for the current `collect digits` command are deleted, digit collection is restarted, and the announcement is not replayed.

In using this command, the maximum number of digits requested of the caller must be specified in the administration of the command. If the caller can enter fewer digits than the maximum specified, the announcement should instruct the caller to terminate the entry with a pound sign (#) digit as an end-of-dialing indicator. If all the digits strings for all the variations of a specific `collect digits` command are terminated with #, the # must be counted as one of the digits. Therefore, the number of digits collected should include any # that needs to be collected. Otherwise, the terminating # is kept as a dial-ahead digit and is processed by a subsequent `collect digits` command. If fewer digits than the maximum specified are entered, and if the caller does not complete the entry with a pound sign, an interdigit timeout occurs. The timeout terminates the command, and any digits collected prior to the timeout are available for subsequent vector processing.

Generally, processing of the command requires that a TTR be connected. (If the call originates from an internal rotary phone, no TTR is needed.) TTRs accept the touch-tone digits that are entered by Call Prompting users. TTRs are automatically connected as needed by the system.

The connection of the announcement prompt is skipped and the digit collection phase begins whenever one of the following conditions is true:

- Dial-ahead digits exist.
- No announcement is administered for the `collect digits` step.
- Announcement administered for the `collect digits` step does not exist.

Otherwise, an attempt is made to connect the administered announcement. If the announcement to be connected is busy, and if the queue for the announcement is full, or if there is no queue, the calling party continues to hear the current feedback. The system waits five seconds and then tries again to connect the call to the announcement. This process continues until the call is successfully queued or connected to the announcement, or until the calling party disconnects from the call. If the queue for the announcement is not full, the call is queued for the announcement.

If the announcement to be connected is available (either initially or after queuing, or after system retry), any previous feedback is disconnected, and the calling party is connected to the announcement.

While the announcement is playing, or while the call is being queued for an announcement, the caller may enter digits at any time. This causes the announcement to be disconnected or removed from the queue, as appropriate, and the digit collection phase to begin. If the caller does not enter any digits during the announcement phases, the digit collection phase begins when the announcement completes.

As soon as the digit collection phase begins, interdigit timing is started, unless the TTR is already in timing mode (that is, the dial-ahead capability is active and the TTR is not disconnected).

Digits are collected either as digits dialed during the **`collect digits`** command or as dial-ahead digits dialed since a previous **`collect digits`** command but prior to the current appearance of the command. Digit collection continues for the current command until one of the following conditions exists:

- Number of digits specified is collected.
- Pound sign (#) digit is collected (signifying end of dialing).
- Inter-digit timer expires.

If, during the digit collection phase, a * is encountered within a stream of dialed or dial-ahead digits, all digits that are collected for the current **`collect digits`** step are discarded. If additional dial-ahead digits occur after the asterisk, these digits continue to be processed. If there are no such digits, and if no TTR is connected, vectoring continues at the next vector step. If a TTR is connected, the caller can start entering digits again. In such a case, the announcement is not replayed, and the interdigit timer is restarted.

Note:

If an asterisk is entered after the requested number of digits are entered, the asterisk has no effect on the previously entered digits. However, in such a case, the asterisk is treated as a dial-ahead digit for the next **`collect digits`** command.

When digit collection is completed, and if a TTR is connected (for a touch-tone phone), the interdigit timer is restarted to detect a timeout for releasing the TTR. Vector processing then continues at the next vector step. However, the switch continues to collect any subsequent dialed digits (including the pound sign (#) and asterisk (*) digits) to allow for the dial-ahead capability. These additional “dialed ahead” digits are saved for use by subsequent **`collect digits`** commands, and they provide the caller with a means to bypass subsequent unwanted announcement prompts. A single # digit can be collected and tested by subsequent **`route-to...if digits`** or **`goto...if digits`** commands. 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 and the dial-ahead digits exceeds the switch storage limit of 24. Any additional dialed digits are discarded until storage is freed up by a subsequent **`collect digits`** command.

Note:

Any asterisk (*) or pound sign (#) digits count towards the 24-digit limit, as do any dial-ahead digits entered after the asterisk or pound sign digit.

- The TTR required by the touch-tone phone user to collect digits is disconnected. This occurs under the following conditions:
  - — Successful or unsuccessful **`route-to number`** step is encountered during vector processing except where the number routed to is a VDN extension.
  - — Successful or unsuccessful **`route-to digits`** step is encountered during vector processing except where the number routed to is a VDN extension.
  - — 10 second timeout occurs, during which time the caller does not dial any digits, asterisks (*) or pound signs (#).

Note:

When the TTR is disconnected due to a **`route-to number`** or **`route-to digits,`** all dial-ahead digits are discarded. This means that, following a failed **`route-to`** step, a subsequent **`collect digits`** step always requires the caller to enter digits.

Note:

Dial-ahead digits are available for use only by subsequent **`collect digits`** commands. The digits are never used by other vector commands that operate on digits (for example, **`route-to digits`**, **`goto...if digits`**, etc.). In addition, these digits are not displayed as part of the CALLR-INFO button operation until they are collected via a **`collect digits`** command.

Answer supervision considerations

Answer supervision is provided as soon as a TTR is connected and processing of the command starts. The command always provides answer supervision to an incoming trunk if supervision has not been previously provided.

Feature interactions

None.

BCMS interactions

Digits are not passed to the BCMS.

For DEFINITY software version 9.1 or later, user adjustments also apply to available agent situations (with a strategy other than first found) in a manner that is similar to EWT. For more information, see the “Best Service Routing” section in: *Avaya Communication Manager Contact Center – Guide to ACD Contact Center*, 555-233-516.

Disconnect command

Command	Parameters / Conditions

Purpose

The **`disconnect`** command ends treatment of a call and removes the call from the switch. Also allows the optional assignment of an announcement that will play immediately before the disconnect.

Important:
You should always warn the caller prior to disconnecting the call.

Syntax and valid entries

Command	Parameters / Options
`disconnect`	**`after announcement`** _____ `extension no. or “none”`

Requirements

The relevant announcements must be administered and recorded.

Operation

While the command’s optional announcement is playing, the call remains in queue and can be connected to an agent. When the announcement completes (or is not specified), the command forces a disconnect, ends the treatment of the call, and removes the call from the switch.

Answer supervision considerations

If the switch has not yet sent answer supervision, the switch does so immediately before disconnecting the call, whether an announcement is specified or not. If an announcement is specified, answer supervision is given before an attempt is made to connect the announcement. The exception is for ISDN calls, where the disconnect can occur without answer supervision being sent when an announcement is not played.

Feature interactions

The command is considered a call acceptance vector command whenever an announcement is included within the command and one of the following is true:

- Announcement is available.
- Call is queued for an announcement.
- Announcement is retried.

The command is considered a call denial vector command whenever one of the following is true:

- No announcement is included within the command.
- Announcement is included within the command, but the announcement is unavailable.

BCMS interactions

A call that is disconnected via the command is tracked as OTHER in the VDN Report.

Goto step and goto vector commands

Purpose

The **`goto step`** command allows conditional or unconditional movement (branching) to a preceding or subsequent step in the vector.

The **`goto vector`** command allows conditional or unconditional movement (branching) to another vector. The goto vector step does not remove a call from queues in which it is already placed.

All parameters, options and value limits are identical for the **`goto step`** and **`goto vector`** commands.

Syntax and valid entries

Command	Parameters / Conditions			
goto **step** _____ 1-32 or **vector** ______ 1 to switch max				
if **available-agents**				
	in split _____ 1 to switch max		____ <,<= >,>=,<>,=	____ 1 to 1500 0 to 1499[1]
calls-queued				
	in split ____ 1 to switch max	**pri** ___ **l**ow-priority **m**edium-priority **h**igh-priority **t**op-priority	____ <,<= >,>=,<>,=	____ 1 to 999[1] 0 to 998[1]
digits ___		_____		
	<,>,=	1-16 digits, including 0-9, “#”, “?”, or “none”[2]		
	=	meet-me-access[3]		

Command	Parameters / Conditions			
goto step (or vector) (*continued*)				
if (*continued*)				
holiday ____ in, not-in	**table ____** 1-10			
meet-me-full[3] **(go-to step, only)**				
meet-me-idle[3] **(go-to step, only)**				
no match[4]				
oldest-call-wait	**in split ____** 1 to switch max	**pri ___** **l**ow-priority **m**edium-priority **h**igh-priority **t**op-priority	____ <,>,=	____ 0 to 999 seconds
queue-fail[5]				
staffed-agents	**in split ______** 1 to switch max		____ <,>,=	____ 1-1500[1]
unconditionally				

1. Maximum limit less on some platforms. Use the help key for your switch administration software to determine the applicable limit for your system.

2. The question mark (?) is a wild card that matches any digit (0-9) at the specified position. The plus sign (+) matches any or no characters at the specified position

3. This item available with **meet-me conference** vectors, only.

4. This item available with **Dial by Name** feature, only.

5. This item available with **Attendant Vectoring** feature, only.

Requirements

For more information about options required to enable the **goto** commands, see Comunication Manager options required to enable vector commands on page 153.

No special requirements.

Operation

If the command syntax includes **unconditionally**, the command always branches. The unconditional form of the command is commonly used for skipping vector commands as well as for looping through vector commands.

Otherwise, branching takes place according to one of the conditions that follow:

- The number of available agents in the indicated split meets the constraints defined by the comparator and the threshold value.
- The number of queued calls in the indicated split and at the specified priority level (or higher) meets the constraints defined by the comparator and the threshold value.
- The number of active calls in the indicated VDN meets the constraints defined by the comparator and the threshold value.
- The oldest call-waiting in the indicated split at the specified priority level (or higher) has been waiting for a period of time within the constraints defined by the comparator and the threshold value, which is expressed in seconds.
- The number of staffed agents in the indicated split meets the constraints defined by the comparator and the threshold value.
- Digits collected via the **`collect digits`** command match the criteria defined by the comparator for the specified digit string. The # digit can be tested against as a single digit.
- Time-of-day criteria are met.

  Note:

  The syntax for this condition can be illustrated by a couple of examples, as follows: **`mon 8:01 to fri 17:00`** means anytime between 8:01 A.M. Monday through 5:00 P.M. Friday, and **`all 17:00 to all 8:00`** means between 5:00 P.M. and 8:00 A.M. on any day of the week.
- For Attendant Vectoring, there is no way to check ahead of time to see if a call can queue, and there is no way to check if, after the fact, a call queued successfully. The **`queue-fail`** command allows you to provide additional routing if a call to an attendant vector fails. You can redirect the call to another step or to another vector if the call cannot be queued.

Answer supervision considerations

The call answer is not affected by the command.

Feature interactions

None.

BCMS interactions

The **`goto step`** and **`goto vector`** commands are not tracked on BCMS.

Messaging command

Purpose

The **`messaging split/skill`** command allows the caller to leave a message for the specified extension or the active or latest VDN extension (default).

Syntax and valid entries

Command	Parameters / Conditions	
messaging		
	split ____ 1-switch max	**for extension** ____ extension no., “latest” or “active”[1]

[1] “Active” refers to the VDN specified by VDN Override settings. “Latest” refers to the VDN specified for the current vector.

Requirements

The split involved must be an AUDIX split.

Operation

This command causes the caller to be connected to the AUDIX or Message Center split so that the caller may leave a message for the specified extension (call answering service or “mail”).

If the split number specified in the command is a valid message service split (such as an AUDIX), and if the extension is either a valid assigned extension or is administered as active or latest the system attempts to terminate the call to the message service split for call answering service.

If the call is queued to the message service split, or if the call terminates to an available message service agent or an AUDIX voice port, the caller is connected to ringback (signifying successful termination), and vector processing terminates. Termination is unsuccessful, and vector processing continues at the next vector step if any one of the following is true:

- Split queue is full.
- AUDIX link is down.
- All AUDIX voice ports are out of service.

If call termination is successful, and if the administered extension (or default VDN) is a message service subscriber, the caller can leave a message for the specified extension.

Note:

> Agent and/or supervisor stations may be equipped with Automatic Message Wait (AMW) lamps to accommodate the “mail” specified in the **`messaging split`** command. The lamps can be assigned for VDNs or extensions used to access the messaging split and for which messages are to be left. When messages are left for these VDNs or extensions, the assigned AMW lamps light.

If the extension or VDN is not a subscriber of the message service, the caller receives ringback until he or she disconnects.

Answer supervision considerations

If answer supervision has not already been returned, it is returned when the messaging service port or station is connected to the call (that is, when the call is answered by the port or station).

Feature interactions

The command can use an AUDIX hunt group in its operation.

If the command specifies a specific “mailbox” extension, the original principal for a call covered by a VDN is not passed to the adjunct, and it does not appear in the display to the answering agent. The specified extension appears in the display.

If the command is accessed via a direct call to the VDN, and if the mailbox is administered as active or latest, the corresponding active or latest VDN extension mailbox is sent to the messaging adjunct. Additionally, if the call is sent to a switch Message Service split/skill, the associated VDN name is sent to the messaging adjunct.

If the command specifies active or latest as the mailbox extension, the original principal for a call covered to or forwarded to a VDN is used as the default mailbox for the call instead of the active or latest VDN. Accordingly, the original principal extension and the reason for redirection are passed to the messaging adjunct, and they subsequently appear in the display to the answering agent.

AUDIX does not support mixed length numbering plans.

If the command leaves a message for a VDN or for another messaging service extension, the Automatic Message Waiting Lamp (AMWL) associated with the VDN or extension lights steady.

BCMS interactions

A call advanced to another position via the command is tracked as an outflow in the VDN Report.

Queue-to command

Purpose

The **`queue-to`** command unconditionally queues a call to a split, attendant group, attendant, or hunt group, and assigns a queuing priority level to the call in case all agents or attendants are busy.

Syntax and valid entries

Command	Parameters / Conditions	
`queue-to`		
	`attd-group`[1]	
	`attd-group`[1]	

	`attendant`[1] `________`
	`extension no.`	

	`hunt-group`[1] `_____`
	`1 to switch max`
		`pri` `___`
		`l``ow-priority`
		`m``edium-priority`
		`h``igh-priority`
		`t``op-priority`

	`split` `____`
	`1 to switch max`
		`pri` `___`
		`l``ow-priority`
		`m``edium-priority`
		`h``igh-priority`
		`t``op-priority`

[1] This item available with Attendant Vectoring feature, only.

Requirements

The split involved must be vector-controlled.

Operation

A call sent with this command either connects to an available agent or attendant in the specified resource or enter the resource’s queue. When it enters the queue, feedback is not given to the caller by this command.

Note:

In Attendant Vectoring, a wait-time 0 secs hearing ringback step should be used to give immediate feedback to the caller. The queue-to command does not provide ringback until the call is actually ringing the attendant. The wait-time step should be implemented as the first vector step or as the step immediately before the queue-to step.

A call may be queued to up to three local split simultaneously. A call remains queued either until vector processing terminates (via a **`disconnect`**, **`busy`**, or **`route-to`** command, or via a dropped or abandoned call) or until the call reaches an agent. When an agent becomes available in any split to which the call is queued, the following actions take place:

- Call begins ringing the agent.
- Call is removed from any other queues.
- Vector processing terminates.

If the entered split is one of the split/skill to which the call is already queued, the call is requeued at the new priority level. If the priority level specified is the same as the priority level at which the call is queued, the call remains in the same position in queue. The step is skipped, and vector processing continues at the next step if any of the following conditions are true:

- Desired split's queue is full.
- Desired split's is not vector-controlled.
- Desired split's has no queue and also no available agents.
- Call has been previously queued to three different splits.

> **Note:**
>
> A **`route-to`** to another VDN can be used to remove the call from the splits it is queued to if necessary. The steps in the routed-to vector then can be used to queue to other splits.

Answer supervision considerations

Answer supervision is returned (if not already returned) when the call is connected to an answering agent.

Feature interactions

The **`queue-to`** command can access an AUDIX split in cases where a VDN is assigned as a coverage point. To enable this function, the split must be assigned as a vector-controlled hunt group.

The command is considered a call acceptance vector command whenever one of the following is true:

- Call terminates to an agent.
- Call queues to a split.

The command is considered a neutral vector command when the call neither terminates nor queues.

No COR checking is carried out when a queue-to step places a call to a split.

BCMS interactions

The total number of calls to the VDN that are queued via the command and then answered by an agent within a specified time period is tracked as ACD Calls in the VDN Report. The average time that calls spend in a vector before being connected via the command as an ACD call to an agent is tracked as AVG SPEED ANS in the same report.

Note:

This multi-site BSR command is available only when the Virtual Routing feature is enabled.

Route-to command

Purpose

Routes calls either to a destination that is specified by digits collected from the caller (**`route-to digits`**), or routes calls to the destination specified by the administered digit string (**`route-to number`**).

Syntax and valid entries

Command	Parameters / Conditions		
route-to			
	digits	**with coverage** _____ **y**es or **n**o	
	meet-me[1]		
	number ____ 0-9, * , # , ~p, ~m, ~s, ~w, ~W, or a leading ~r[2]	**with cov** ___ **y**es or **n**o	**if** ____ digit unconditionally
	name1 [3]	**with coverage** _____ **y**es or **n**o	
	name2 [3]	**with coverage** _____ **y**es or **n**o	
	name3 [3]	**with coverage** _____ **y**es or **n**o	

[1] This item available with **`meet-me conference`** vectors, only.

[2] When the specified number is preceded by **`~r`**, Network Call Redirection is attempted.

[3] This item available with Dial by Name feature, only.

Requirements

The **`route-to name`** command requires the Dial by Name feature to be enabled.

Operation

The **`route-to`** command attempts to route a call to a set of digits collected from the caller, from an adjunct, or from the network, or to route to the destination specified by the administered digit string.

For the **`route-to number ... if digit`** command, the call is conditionally routed to a specified destination according to a single digit entered by the caller. If the digit collected in the last **`collect digits`** command matches the specified comparison in relation to the administered digit, the command attempts to route the call to the specified destination.

The destination for a route-to command can be any of the following:

- Internal extension (for example, split/hunt group, station, etc.)
- VDN extension
- Attendant or Attendant Queue
- Remote extension (UDP)
- External number, such as a TAC or AAR/ARS FAC followed by a public or private network number (for example, 7-digit ETN, 10-digit DDD, etc.)
- Remote Access Extension.

Note:

The VDN's Class of Restriction (COR) is used for calling permissions.

The **`route-to digits`** command fails if no digits are collected, and vector processing continues at the next vector step.

The **`route-to number ... if digit`** command fails if more than 1 digit is collected or if the digit comparison fails. Vector processing continues at the next command.

If the **`route-to`** command is successful, vector processing terminates. Otherwise, vector processing continues at the next vector command.

A **`route-to`** step in a vector is treated as cov=n for a covered call regardless of the cov setting on the **`route-to`** command.

If the number expressed in the command is a system extension or an attendant group (and not a VDN), the system considers the step successful if one of the following conditions occurs:

- The endpoint is alerted.
- The endpoint has Call Forwarding or night service (hunt group) enabled, and the (night service) destination forwarded to is alerted.
- The endpoint has off-premises Call Forwarding (UDP hunt night service) enabled, and a trunk is seized.

The system then provides ringback to the caller, and vector processing terminates. However, if the call cannot complete successfully (for example, no idle appearance is available), vector processing continues at the next vector command.

If the number is a VDN extension, the following events occur:

- Vector processing terminates within the current vector.
- If the current VDN is administered with override, the new VDN overrides current VDN information.
- Processing of the vector associated with the VDN extension begins.

If the number is an AAR/ARS FAC plus digits, or if it is a remote UDP extension, standard AAR/ARS processing is performed to select the trunk group and outpulse the digits. If a trunk is seized, vector processing terminates, and the calling party hears feedback provided by the far end. Otherwise, the call cannot complete successfully (because no trunks are available, the FRL/COR is restricted, etc.), and vector processing continues at the next vector command.

If the number is a TAC plus digits, and a trunk is seized, vector processing terminates, and the calling party hears feedback provided by the far end. Otherwise, the call cannot complete successfully (because no trunks are available, the COR is restricted, etc.), and vector processing continues at the next vector command.

If the number is any other number (such as an FAC other than an AAR/ARS), the command is unsuccessful, and vector processing continues at the next vector command.

Abbreviated Dialing special characters can also be used in the number field. Each of these characters instructs the system to take a different action when dialing reaches the point where the character is stored. The characters are as follows:

- **`~p`** (pause)
- **`~w`** (wait)
- **`~m`** (mark)
- **`~s`** (suppress)
- **`~W`** (indefinite wait)

Each special character counts as two digits towards the maximum. The maximum number of digits for the command is 16.

The **`route-to digits`** command can be used to implement an automated attendant function.

Coverage

The optional coverage parameter determines whether coverage should apply during routing. If coverage applies, and if the digits entered are valid, the following occurs:

- Ringback is provided.
- Vector processing terminates.
- Normal termination and coverage are implemented.

Note:

For detailed information about the operation of the route-to command with or without coverage for the different destinations see the table shown in Switch route-to command operation on page 227.

Answer Supervision Considerations

Generally, answer supervision is provided when the destination answers the call. The exception to this involves incoming trunk calls routed to another non-ISDN-PRI trunk. Such calls provide answer supervision when the outgoing trunk is seized.

Feature interactions

When COR checking is applied to a route-to number or route-to digits step, it is the COR of the latest VDN that is used.

The **`route-to`** command may specify the AAR or ARS access codes. The COR associated with the latest VDN is used to determine the Partitioned Group Number (PGN) time-of-day routing chart. The PGN determines the choice or route tables used on a particular call.

The command may call the AUDIX extension. If this happens, the call is treated as a direct call to AUDIX, and the calling party may retrieve his or her messages.

If the call covers to a VDN, the command supports a remote AUDIX interface to a local hunt group extension that is assigned as a remote AUDIX hunt group. The remote AUDIX hunt group (which has no members and cannot be vector-controlled) forwards the call to the remote AUDIX destination in the same manner as when the hunt group is assigned as a point in the coverage path.

If the command is directed to a station with bridged appearances, the bridged appearance button lamps are updated.

The following destinations always result in a failure, and vector processing continues at the next step:

- Controlled trunk group
- Code calling FAC
- Facility test call
- TAAS access code
- Priority access code
- Loudspeaker paging access code
- Station Message Detail Recording (SMDR) account code
- Voice message retrieval access code.

If the command is executed and Direct Outward Dialing (DOD) is in effect, the COR of the latest VDN is compared with the COR of the called facility to determine if the call is permitted. If access is not permitted, the command fails and vector processing continues. In the case where a COR requiring the entry of account codes is assigned to a VDN, and the command is executed by the associated vector, the command is unsuccessful, and vector processing continues at the next step.

The individual extension number assigned to an attendant console can be used as the command's argument.

A call processed by the command can wait in the individual attendant queue and is subsequently removed from vector processing.

The command can access both public and private networks.

If the command dials the attendant, and if the system is in night service, the call routes to the DID Listed Directory Number (LDN) night destination.

The command can place AAR/ARS calls that implement subnet trunking, which is the routing of calls over trunk groups that terminate in switches with different dial plans.

Authorization codes are disabled with respect to routing via VDNs. In other words, if authorization codes are enabled, and a **`route-to`** command in a prompting vector accesses AAR or ARS, and the VDN's FRL does not have the permission to utilize the chosen routing preference, no authorization code is prompted for, and the **`route-to`** command fails.

If the command routes the call without coverage to a display station, the station displays the following: "a = Originator Name to VDN Name."

If the command calls a station that is a member of a pickup group, the call can be picked up by another pickup group member.

The command is considered a neutral vector command whenever one of the following is true:

- Termination is unsuccessful.
- Trunk is not seized.

For a call that covers or forwards to a VDN, the **`route-to with coverage y`** command functions the same way as the **`route-to with coverage n`** command. For a covered or forwarded call, the coverage option for the command is disabled since such a call should not be further redirected.

A **`route-to with cov y`** to a station that has call forwarding activated is forwarded.

Service Observing can be initiated with Call Vectoring using the route-to command.

Note:

Appendix F: Operation details for the route-to command on page 227 gives a detailed description of the feature interactions for the **`route-to`** number with and without coverage command.

BCMS interactions

A call advanced to another position via the command is tracked as outflow in the VDN Report. A call answered by an attendant via the command is also tracked as outflow.

Stop command

Purpose

The **`stop`** command halts the processing of any subsequent vector steps.

Syntax

```
stop
```

Requirements

No special requirements.

Operation

After the **`stop`** command is processed, any calls already queued remain queued, and any wait treatment (for example, silence, ringback, music) is continued. On the other hand, any calls not queued are dropped under the same scenario.

If a TTR is allocated to the call, and if the **`stop`** command is encountered, the TTR is disconnected. However, current call processing continues (that is, the call is not dropped). The caller continues to hear the feedback that was provided before the **`stop`** command was encountered.

Note:

An implicit stop is processed following the last administered command in a vector.

Answer supervision considerations

The command has no effect on answer supervision.

Feature interactions

None.

BCMS interactions

None.

Wait-time command

Purpose

The **`wait-time`** command delays the processing of the next vector step if a specified delay time is included in the command's syntax. Also provides feedback (in the form of silence, ringback, or music) to the caller while the call advances in queue. The Multiple Audio/Music Sources for Vector Delay and Multiple Music Sources on Hold features allow a specified audio or music source to be selected when a call encounters a **`wait-time`** command. See Basic Call Vectoring for more information.

Syntax and valid entries

Command	Parameters / Conditions			
`wait-time`	____	____	**`hearing`**	_____________
	`0-999`	`secs`		`music, ringback, silence`

Requirements

Basic Call Vectoring or Call Prompting software must be installed. Also, a music-on-hold port must be provided for the music treatment.

Operation

The specified feedback is given to the caller, and vector processing waits the specified time before going on to the next step. If the time specified is **`0`**, feedback is provided without any delay in the processing of the next vector step. The feedback given to the caller continues until any one of the following occurs:

- Subsequent vector step (containing **`wait-time`** or **`announcement`**) changes the treatment.
- Vector processing encounters a **`disconnect`** or **`busy`** command.
- Call is routed to another location or to a step that includes an announcement (for example, **`collect digits`**).
- Call is routed to another VDN.
- Call is delivered to a destination (starts ringing at an agent's terminal).

Considerations

When music is indicated as a treatment, it refers to the system music, not an alternate music source.

The tenant number of the active VDN determines the system music the caller hears. You can allow callers to hear a music source other than the one assigned to the active VDN, however, by directly specifying an extension for an audio source with a command such as **`wait-time 30 secs hearing 4301 then music`**.

Multiple audio/music sources

The expanded **`wait-time _ secs hearing <extension> then <treatment2>`** command provides what is known as Multiple Audio/Music Sources wait treatment. The **`<extension>`** option defines an audio or music source that is assigned on the Announcements/Audio Source administration screen.

The source can be interfaced by way of one of the following:

- Analog/DS1/0 (Line Side T1/E1) station ports
- AUX-Trunks
- An Integrated Announcement board

Any of the announcement/audio source types listed above can be configured to do either of the following:

- Play at the beginning with queuing (with the `Queue` field set to y, which is always recommended for contact center applications)
- Barge-in operation (Queue field set to "b")

In addition, integrated board announcements can be set to play once ("integrated") or to repeat after each playing continuously (integ-rep). For more information, see “Appendix A: Recorded Announcements”, in *Avaya Communication Manager Contact Center – Guide to ACD Contact Center*, 555-233-516, and the Announcements/Audio Sources screen reference in *Administrator Guide for Avaya Communication Manager*, 555-233-506.

The **`<treatment2>`** parameter refers to the treatment that the caller hears after the source specified by **`<extension>`** finishes playing, or the wait-time period expires. The **`<treatment2>`** parameter is also provided if the caller can not be connected to the source. Failure to connect to the source can result from conditions such as:

- source not available - extension/source not assigned
- source disconnected
- source busy
- queuing not assigned

If the **`<extension>`** source is not available when the wait step is reached in the vector one of the following results will occur:

- If **`<treatment2>`** is set to **`continue`**, the caller returns to what they were hearing before the wait-time step.
- If **`<treatment2>`** is set to **`music`**, **`ringback`**, or **`silence`**, vector processing still waits for the specified wait-time while the caller hears **`<treatment2>`**. When the wait-time period expires, the next step in the vector is executed, regardless of the **`<treatment2>`** setting. The caller continues to hear **`<treatment2>`** until a subsequent step changes the treatment. For example, if **`<treatment2>`** is set to **`continue`**, and the **`<extension>`** source (integ-rep or continuous analog/DS1 or AUX-Trunk) is still playing, the caller continues to hear it until a subsequent vector steps changes the treatment.

Note:

If the **`<extension>`** source stops playing or is disconnected, the caller hears silence.

If the audio/music source specified by the **`<extension>`** stops (disconnects) before the wait-time period expires or the caller cannot be connected to that source (source not available), the caller will hear the source specified by the **`then <treatment2>`** segment of the vector. In this case, if **`<treatment2>`** is specified as **`continue`**, then the caller hears silence.

Answer supervision

If the **`music`** or **`audio source`** treatment is included in the command, answer supervision is triggered. If the command is encountered and answer supervision was sent previously, the caller hears the treatment specified in the current command. If, for a CO trunk user, the command with **`silence`** or **`ringback`** treatment is encountered prior to answer supervision, the caller continues to hear ringback from the CO.

Feature interactions

An implicit wait of 0.2 seconds (with no change in the feedback to the caller) is provided after every seven vector steps if one of these steps does not suspend vector processing. The following steps, if successful, do not suspend vector processing: **`queue-to split`**, **`check split`**, **`goto step`**, **`goto vector`** and **`wait-time 0 seconds`**. The following steps, if unsuccessful, also do not suspend vector processing: **`check split`**, **`route-to`**, and **`messaging split`**. The only commands that suspend vector processing are the following: **`announcement`**, **`wait-time > 0,`** and **`collect digits`**.

- Music-on-Hold

  When the command is implemented with music as the treatment, the system-wide music-on-hold feature must be administered. Otherwise, the caller hears silence. When Tenant Partitioning is in use, the tenant number of the active VDN determines the system music that is heard.

  Feedback continues while a subsequent vector step queues for an announcement or for a TTR.

BCMS interactions

The command is not tracked on BCMS.

Appendix B: Vector management and monitoring

To manage your vectors, there are several considerations and tasks of which you need to be aware. This appendix describes these considerations/tasks, including:

- Implementation requirements for the Call Vectoring features on page 191
- Enabling the Vector Disconnect Timer on page 193
- Changing and testing a vector on page 194

Implementation requirements for the Call Vectoring features

The following tables indicate the forms and the hardware required for the following Call Vectoring features:

- Basic Call Vectoring Requirements on page 191
- Call Prompting Requirements on page 192

Basic Call Vectoring Requirements

Form(s)	Hardware
• Vector Directory Number • Hunt Group • Call Vector • Feature Related System Parameters	Announcement capabilities require either: • TN750 Integrated Announcement circuit pack(s), or • External announcement facility (analog announcements). Also, each analog announcement requires a port on an analog line circuit pack or on an auxiliary trunk circuit pack. See the *Hardware Guide for Avaya Communication Manager*, 555-233-200 for a list of available analog circuit packs.

Note:

The TN750 Integrated Announcement circuit pack provides 16 ports for listening to announcements. The system provides for the installation of multiple TN750C Integrated Announcement circuit packs. See “Managing Announcements” in *Administrator Guide for Avaya Communication Manager*, 555-233-506 for more details.

Call Prompting Requirements

Form(s)	Hardware
● Vector Directory Number ● Hunt Group ● Call Vector	Announcement capabilities require either: ● TN750 Integrated Announcement circuit pack(s), or ● External announcement facility (analog announcements). Also, each analog announcement requires a port on an analog line circuit pack. See *Hardware Guide for Avaya Communication Manager*, 555-233-200 for a list of available analog circuit packs.

Enabling the Vector Disconnect Timer

Call Vectoring provides a Vector Disconnect Timer, which can be set for any amount of time between 1 and 240 minutes inclusive. The timer is enabled by selecting the timer field in the Feature-Related System-Parameters form. The timer is started when vector processing is started. Once the timer runs out, the call is dropped. The timer is canceled when vector processing terminates.

Enabling the timer allows queued calls that have not been answered within a determined amount of time to be dropped. For more information, see *Avaya Communication Manager Contact Center – Guide to ACD Contact Center*, 555-233-516.

Changing and testing a vector

Vectors currently being used to process calls should not be changed because changes would have an immediate and uncertain effect on the treatment that the calls are receiving. Instead, a new vector should always be written.

In testing the vector, you should not consider the entire vector at once. Rather, you should first figuratively divide the vector into portions, then test each of these portions until the entire vector is tested.

After the new vector is thoroughly tested, the vector should be brought into service by changing the VDN to point to the new vector.

The set of following guidelines is intended to serve as a general procedure for changing and testing vectors.

1. Check that a current version of the translation data is available.
2. Create a new VDN that points to the new vector. This VDN, which is temporary, is necessary to test the new vector.
3. Administer the new vector. Vector commands should be added and tested, one command at a time, starting with the first command. Be sure that each line is correct before proceeding to the next one.
4. Test the new vector with the new VDN. This ensures the new vector will function correctly when the vector is installed.
5. Install the new vector by changing the old VDN's vector assignment so that the VDNs now point to the new vector. Calls that are already being processed by the old vector will continue to be handled by that vector until the vector terminates vector processing.
6. Once all the calls are handled, remove the old vector and the VDN that was used for testing.

```
list usage vector 3                                                  Page 1

                        LIST USAGE REPORT

Used By
Vector                   Vector Number          1                 Step 3
VDN                      VDN Number             58883
```

Appendix C: Considerations for the vectoring features

This appendix provides various considerations you should bear in mind when using the Call Vectoring features. These considerations are intended to help you get the highest degree of productivity from Call Vectoring.

This appendix includes the following topics:

Transferring calls 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. 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 features not supported

The following Call Vectoring features are not supported with BCS or GuestWorks:

- Adjunct Routing
- Advanced Vector Routing
- ASAI Routing (not the same as Adjunct Routing)
- ANI/II Digits Routing
- Best Service Routing
- Call Information Forwarding (CINFO)
- Expert Agent Selection
- G3V4 Enhanced Features
- Look-Ahead Interflow
- VDN of Origin Announcements

Call Vectoring commands not supported

The following vectoring commands (vector steps) are not available with BCS or GuestWorks:

- adjunct routing
- check skill
- check best
- consider split
- consider skill
- consider best
- converse-on split
- converse-on skill
- messaging skill
- queue-to skill
- queue-to best
- reply-best

Appendix D: Troubleshooting vectors

This appendix serves as a troubleshooting guide for Call Vectoring. The sections include the following topics:

Criteria for success/failure of call vectoring commands

The following table summarizes the success and failure criteria for various vector commands. Before you write or evaluate vectors, it is important to understand the information in this table.

Call vectoring command success/failure criteria	
`announcement`	
Fails if specified announcement is not administered, not recorded, or busied out.	Continue vector processing with the next sequential step.
Otherwise, succeeds.	Play the announcement, then continue at the next sequential step.
`busy`	
Always succeeds. (CO without answer supervision trunk callers will not hear the busy tone.)	Exit vector processing, then play the busy tone for 45 seconds before dropping the call. (Unanswered CO trunk calls receive 45 seconds of ringback.)
`check split`	
Fails if any of the following are true: ● Vector conditional is false. ● Split's queue is full. ● Split is not vector-controlled. ● Call is already queued at the specified priority to the specified split. ● Call is already queued to three different splits.	Continue vector processing with the next sequential step.
Otherwise:	
Succeeds, and the call is terminated to an agent.	Exit vector processing, and pass control to call processing.
Succeeds, and the call is queued or requeued in the specified split at the specified priority.	Continue vector processing with the next sequential step.

Call vectoring command success/failure criteria (Continued)	
`collect-digits`	
Fails if any of the following are true:	
Call originates from an outside caller who is not using a touch-tone telephone.	Call Prompting timer takes effect, command times out, and vector processing continues at the next vector step.
No TTR is in the system, or the TTR queue is full.	Continue vector processing at the next step.
Caller enters fewer digits than the maximum specified.	Call Prompting timer takes effect, command is terminated, and any digits collected prior to the timeout are available for subsequent processing.
Otherwise, succeeds.	Continue vector processing at the next step.
`disconnect`	
Always succeeds.	Play the announcement (if specified). Then drop the call.
`goto step and goto vector`	
Fails if the step condition is not met.	Continue vector processing with the next sequential step.
Succeeds if the step condition is met.	**`goto step`** - continue vector processing with the destination step **`goto vector -`** continue vector processing with the first nonblank step of the destination vector.

Call vectoring command success/failure criteria (Continued)	
`messaging split`	
Fails if any of the following are true: ● Specified split is not an AUDIX split. ● Specified extension is invalid. ● Messaging split queue is full. ● Messaging split is not vector controlled and has no working agents (none logged in or all in AUX work mode). ● Communications link with the messaging adjunct is inaccessible.	Continue vector processing with the next sequential step.
Otherwise, succeeds.	Terminate vector processing.
`queue-to split`	
Fails if any of the following are true: ● Split's queue is full. ● Split is not vector-controlled. ● Call is already queued at the specified priority to the specified split. ● Call is already queued to three different splits.	Continue vector processing with the next sequential step.
Otherwise:	
Succeeds, and the call is terminated to an agent.	Exit vector processing, and pass control to call processing.
Succeeds, and the call is queued or requeued in the specified split at the specified priority.	Continue vector processing with the next sequential step.
`stop`	
Always succeeds.	Exit vector processing. Control is passed to normal call processing. Any queuing or treatment in effect remains in effect. Call is dropped if not queued.
`wait-time`	
Always succeeds.	Connect the specified treatment and pass control to the delay timer. Any feedback is continued until other feedback is provided.

Unexpected feature operations

The following table indicates and explains unexpected operations within Call Vectoring that you may encounter.

Unexpected feature operations	
Customer observations	**Causes**
General Vector Processing	
Vector stuck	1000 steps executed. No default treatment in the vector.
Audible feedback lasts longer than the delay interval.	Last vector step. Queuing for an announcement. Queuing for a touch-tone receiver for a **`collect digits`** step.

Unexpected command operations

The following table indicates and explains the unexpected operations the customer may encounter in using the Call Vectoring commands.

Unexpected command operations	
Customer observation	**Cause**
`announcement`	
Announcement not heard.	Announcement board not present. Announcement not administered. Announcement not recorded. Announcement being rerecorded. All ports busied out. Announcement restore in progress. Link to announcement circuit pack down.
Extra delay before hearing announcement.	Announcement queue full. All integrated announcement ports busy. Analog announcement busy.
Vector processing stops.	Analog announcement does not answer.
Listening to silence after announcement.	Announcement is the last step.
Incomplete announcement.	Agent becomes available. Previous adjunct routing step succeeds.
busy	
Ringback heard instead of busy tone.	Unanswered CO trunk.
check	

Unexpected command operations (Continued)

Customer observation	Cause
Call does not enter queue or terminate to agent.	Step condition not met.
`check` and **`queue-to`**	
Call does not enter queue or terminate to agent.	Queue length specified on the hunt group screen has been exceeded. Invalid split. Split not vector-controlled. Already queued to three different splits. No queue. Queue or check status indicates space when queue is full due to direct agent calls.
Call apparently answered in wrong order.	Call being requeued at different priority. Call superseded by higher priority call, including direct agent call.
`collect digits`	
Announcement not heard while waiting for digits, but network billing indicates that the call was answered.	Announcement board not present. Announcement not administered. Announcement not recorded. Announcement being rerecorded. All ports busied out. Announcement restore in progress. Dial ahead digit exists.

Unexpected command operations (Continued)	
Customer observation	**Cause**
`collect digits` (continued)	
Collect step and announcement skipped.	TTR not in system. Link to PN that has TTR is down. TTR queue full.
Delay before hearing announcement.	All TTR ports busy, but space in queue. Announcement queue full. All integrated announcement ports busy. Analog announcement busy.
Vector stuck.	Analog announcement does not answer.
Dial-ahead digits not recognized.	Dial-ahead digits entered prior to first collection step. Call has been transferred. TTR has been released. 24 digits have already been provided. Call Prompting timeout since the last digit was entered.
Vector processing halted at collect step; announcement heard again upon return.	Call put on hold, transferred, or conferenced.
Insufficient digits collected; call routed to intercept.	Caller dialed # too soon. Caller dialed * without reentering correct digits. Call Prompting interdigit time-out.
Caller information button denied.	No digits were collected. Display not in Normal mode.
Collect announcement not heard and first collected digit incorrect.	System does not contain all TN748C Vintage 5 (or later) circuit packs.

Unexpected command operations (Continued)	
Customer observation	**Cause**
Incomplete announcement.	Agent becomes available. First digit dialed.
`disconnect`	
Announcement not heard.	Announcement board not present. Announcement not administered. Announcement not recorded. Announcement being rerecorded. All ports busied out. Announcement restore in progress.
`disconnect` (continued)	
Extra delay.	Announcement queue full. All integrated announcement ports busy. All analog announcements busy.
Vector stuck.	Analog announcement does not answer.
`goto step`	
Branch is not made to the specified step.	Step condition not met. System time not set.
`goto vector`	
Branch is not made to the specified vector.	Step condition not met.
Vector stuck.	Goto vector with no steps or with all failed steps.
`messaging split`	
Vector stuck (with ringback).	Extension unknown to AUDIX.
Step skipped, no message left.	AUDIX link down. Queue for AUDIX voice ports is full.

Unexpected command operations (Continued)	
Customer observation	**Cause**
Vector stuck (with busy).	Remote AUDIX link down.
Messages not found.	Message extension is none (message is left for VDN that accessed the vector).
Delay before AUDIX answers.	All AUDIX ports busy, but space in queue.
Busy tone.	Queue for AUDIX voice ports is full.
Step skipped.	Split not AUDIX split anymore.
`route-to`[1]	
Step skipped (that is, default treatment).	Invalid local extension. No trunks available. COR/FRL restricted. Digit string inconsistent with networking translation. Busy local destination (route to digits without coverage and route to number). No digits collected. Step condition not met.
Network reorder.	Digit string inconsistent with public network translation.
Intercept or reorder tone heard.	Vector processing succeeded routing off switch, but a problem has occurred before routing to its final destination.
All trunks busy on a quiet system.	Two switches treating each other as a backup switch.
`stop`	
Call dropped.	Call not queued when vector processing stops.

Unexpected command operations (Continued)	
Customer observation	**Cause**
`wait-time`	
Audible feedback longer than delay interval.	Queuing for an announcement or for a TTR. **`Stop`** command executed.
Audible feedback shorter than delay interval.	Agent becomes available.
Music not heard.	No music port administered. Music source disconnected or turned off.
Alternate audio/music source not heard	Announcement board not present. Audio/Music source not administered. Audio/Music source not recorded. Audio/Music source being rerecorded. All ports busied out. Announcement restore in progress.

[1] Complete operation details for the route to commands are presented in Appendix F: Operation details for the route-to command on page 227.

Tracking unexpected events

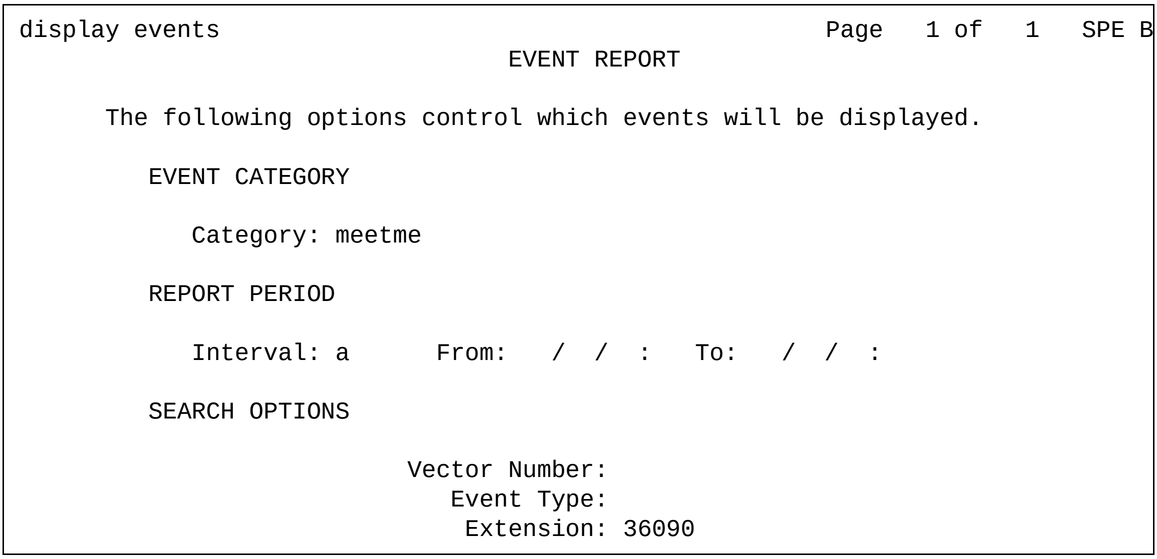
You can display unexpected events related to Call Vectoring and Meet-me Conference. When you have corrected each problem, then you can clear events from the error log. An event is an error that results from resource exhaustion, from faulty vector programming, or from incorrect user operation rather than from a switch software error. For example, failures involving the **`route-to`** command are usually due to an invalid extension entered by the user.

By displaying events, you can diagnose and correct each problem, as indicated by its corresponding event number, and eliminate the need for a technician to make on-site visits to do the same.

The following sections explain how you can troubleshoot by tracking unexpected events.

Display events criteria

Use the **`display events`** command to access the EVENT REPORT screen. Use the fields on this screen to specify the event report criteria.



```
display events                                          Page   1 of   1   SPE B
                                  EVENT REPORT

      The following options control which events will be displayed.

         EVENT CATEGORY

            Category: meetme

         REPORT PERIOD

            Interval: a      From:   /  /  :   To:   /  /  :

         SEARCH OPTIONS

                           Vector Number:
                              Event Type:
                               Extension: 36090
```

The following table describes the fields used with the **`display events`** command.

Field	Description
Category	Enter **`denial`**, **`meetme`**, **`vector`**, or **`all`** to specify the type of event you want to display.
Interval	Select the time period for which you want to display events. Enter **`h`** (hour), **`d`** (day), **`w`** (week), **`m`** (month), or **`a`** (all).
From/To	Enter the date and time of day when you want to start and end the search.
Vector Number	Enter a specific vector number to report on. When the Category field is set to `meetme`, this field is ignored.
Event Type	Enter a specific event type to report on. If this field is blank, events for all types are reported.
Extension	Enter a specific extension or VDN to report on. If this field is blank, events for all extensions are reported.

Display events report

After you have entered your report criteria, submit the command by pressing **Enter**. The following screen shows examples of events.

```
display events

                                 EVENTS REPORT

Event Event                      Event     Event      First        Last     Evnt
Type  Description                Data 1    Data 2     Occur        Occur     Cnt

90    Wait step music failed      3/1     2A2      02/12/15:42 02/13/09:40  255
112   Converse no prompt digits   3/2     2A2      02/12/15:42 02/13/09:40  255
56    Call not in queue           8/1     28B      02/12/15:43 02/13/09:40  255
220   EWT call not queued         8/2     28B      02/12/15:43 02/13/09:40  255
150   Invalid hunt group          8/3     28B      02/12/15:43 02/13/09:40  255
56    Call not in queue           8/5     28B      02/12/15:43 02/13/09:40  255
```

The following table describes the information displayed in the event report.

Column	Description
Event Type	Displays a unique number that identifies the type of event that occurred. These are explained in more detail in Summary of events on page 213.
Event Description	Displays a brief explanation of the event.
Event Data 1	Displays the following data: ● *<number1>/<number2>* (for example, `12/5`), where *<number1>* is the vector number associated with the vector event, and where *<number2>* is the step number associated with the vector event. ● *Split<number>* (for example, `Split 89`), where *<number>* is the split number associated with the vector event. ● For Meet-me Conference events, this is the port ID of the user associated with the event.
Event Data 2	Displays the following data: ● Additional data encoded as a hex number (for example, `4C`). This number serves as a call identifier. If two or more events with an identical identifier occur at about the same time, it can be concluded that the events were caused by the same call. ● For Meet-me Conference events, this is the VDN of the Meet-me Conference used during the event.
First Occur/Last Occur	Displays the date and time the event first occurred and the date and time the event last occurred.
Evnt Cnt	Displays, up to 255, the total number of vector events of this type that have occurred.

Summary of events

The following table provides a list of events, the brief description that displays on the screen, and a full explanation of the event.

Summary of events

Event type	Event description	Explanation
1	Call dropped; call not queued at stop step.	Vector processing ended without the call being queued to a split and, as a result, the call cannot be answered. This implies that some default condition was not programmed or that the vector was designed to not always answer the call. Also, call was subsequently dropped.
2	Vector with no steps	The call encountered a vector with no steps administered.
3	1000 step executed	This can occur due to the following: Incorrect vector programming (for example, including a series of **`goto`** steps that point to one another) Excessive repetition of a programmed loop during a single call (for example, recurring announcement-wait loop)
4	Administration change	The administration of this step occurred while the step was being executed. The call flow for this call is unpredictable. Vectors should not be changed while calls are active.
5	Call dropped by vector disconnect timer	The call was still in vector processing when the vector disconnect timer expired. The call dropped.
7	vec_act.c (07)	There is a mismatch between Attendant Vectoring and Call Vectoring between the VDN and the vector.
8	vec_act.c (08)	There is a mismatch between Attendant Vectoring and Call Vectoring between the incoming call and the VDN.
9	vec_act.c (09)	There is a mismatch between Attendant Vectoring and Call Vectoring between the incoming call and the vector.
10	Retrying announcement	During an **`announcement`** step, a **`collect digits`** step that contains an announcement, or a **`disconnect`** step, the announcement was not available, and the announcement queue (if specified) was full. The step is retried at regular intervals.

Summary of events (Continued)

Event type	Event description	Explanation
11	No announcement available	During an **`announcement`** step, a **`collect digits`** step that contains an announcement, or a **`disconnect`** step, the announcement was not available for one of the following reasons: ● Announcement was not recorded ● Analog announcement was busied out ● Integrated announcement board was not installed ● Integrated announcement ports were busied out ● Integrated announcement was being recorded or restored
20	Call cannot be queued	A **`queue-to split`**, **`messaging split`**, or **`check split`** command failed to queue the call. NOTE: Event types 520, 521, 522 and 541 may be observed for the same call at the same time.
21	Queued to three splits	The call attempted to queue to four splits. Multiple split queuing allows the call to queue to a maximum of three splits simultaneously. If the call queued to one or more splits, and if it should now be dequeued from those splits and then queued elsewhere, one solution is to route the call to a station (which may be administered without hardware). Once this happens, the call is forwarded to the VDN that controls the next stage of the call.
22	vec_q_attd ( ), vect_q_atg ( ), vec_q_hunt ( ) (22)	Applies to Attendant Vectoring and indicates that the call is in the attendant queue and another attempt is made to queue the call to an attendant or hunt group, or the call is in the hunt group queue and an attempt is made to queue it to an attendant or too many attempts are made at queueing to the hunt group.
30	No TTR available	A **`collect digits`** command failed because: ● TN744 port was not available ● All queue slots were occupied
31	Dial-ahead discarded	Previously entered dial-ahead digits have been discarded via access of a(n) **`route-to number`**, or **`messaging split`** step.

Summary of events (Continued)

Event type	Event description	Explanation
32	Prompting buffer overflow	The prompting digit buffer already contained the maximum of 24 digits when additional dial-ahead digits were entered by the caller. These additional digits are not stored.
40	Messaging step failed	A messaging step failed because the Messaging Adjunct was not available. NOTE: Event types 540 and 541 may be observed for the same call at the same time.
50	Route -to step failed	A **`route-to`** step failed to reach the intended destination. NOTE: Event types 51 and 52 may provide more specific information regarding the reason for the failure. See Appendix F: Operation details for the route-to command on page 227.
51	No digits to route-to	The **`route-to digits`** step was unable to route the call because the previous **`collect digits`** step failed to collect any digits. This could result from an error in vector programming (for example, a **`route-to digits`** step appears without a preceding **`collect digits`** step). More often, however, this results because the caller was unable to enter the required digits (that is, the caller was using a rotary telephone), or because the caller was not provided with enough information to do so (as can be the case for auto-attendant applications).
52	No available trunks	A **`route-to`** command was unable to reach the specified off-switch destination due to a lack of available trunks.
53	Route-to step failed	The step was unable to seize a trunk because of a hardware problem or glare.
55	Double coverage attempt	Coverage option on route-to step was ignored because double coverage is not allowed. This may happen when the call has covered to a VDN.
70	Busy step for CO trunk	A CO trunk call reached a **`busy`** step in a vector without having previously received answer supervision. As a result, the caller continues to hear ringback rather than the busy tone.
80	Time not set	A **`goto`** step with a **`time-of-day`** conditional was processed, but the switch time was not set.

Summary of events (Continued)

Event type	Event description	Explanation
81	No digits collected	No digits were collected and a comparison was requested against a digit string or in-table. The comparison test was considered false and the next step in the vector was executed.
90	Wait step music failed	A **`wait-time`** step with music was accessed, but the music was not connected. Music may not be administered correctly.
91	Wait step ringback failed	A **`wait-time`** step with ringback was accessed, but the ringback was not connected.
100	Redirect unanswered call	The call was sent to an agent via a vector, but, due to the Redirection on No Answer (RONA) feature, the call was redirected from the ringing agent.
101	Redirect of call failed	The call was sent to an agent via a vector, but, due to the Redirection on No Answer (RONA) feature, the call was redirected from the ringing agent. The call could not be redirected.
140	Coverage conference denied	Coverage to a VDN in a coverage path was denied because more than one party was active on the call.
222	System clock change	The system clock was changed, therefore any calculations involving time will be inaccurate.
520	Split queue is full	A **`queue-to split`**, **`check split`**, or **`messaging split`** command was executed, but the call did not queue to the split because the queue (if administered) was full. To prevent this condition, use a **`goto step...if calls queued in split...>...`** before each **`queue-to split`** or **`check split`** step so that an alternative treatment may be provided for these cases.
521	Not vector-controlled	The split accessed by a **`queue-to split`** or **`check split`** command is not vector-controlled. As a result, the step is skipped.
522	AAS split cannot queue	A **`queue-to split`**, **`check split`**, or **`messaging split`** command was executed on an auto-available split (AAS), but the call did not queue to the split because all the agents were logged out by Redirection on No Answer (RONA).
540	AUDIX link down	AUDIX could not be accessed via a **`messaging split`** command, because the AUDIX link was down. As a result, the step is skipped.

Summary of events (Continued)

Event type	Event description	Explanation
541	Not a messaging split	The split administered for the **`messaging split`** command is not a messaging split (that is, it does not have a messaging type administered). As a result, the step is skipped.
542	Can't connect idle agent	The call at the head of the queue can't be connected to an idle agent.
1760		
3201	Meet-Me Access chg TMO	The user changing the access code allowed the call to timeout to intercept treatment. The access code was not changed.
3202	Invld Num Digits MM Acc.	The user changing the access code entered too many digits. The access code was not changed.
3203	MM Extension not valid	The user changing the access code did not enter a valid extension.
3204	MM Access Chg Not a VDN	The user changing the access code entered a non Meet-me Conference VDN extension.
3205	MM Invalid Access Entered	The user changing the access code did not enter the correct access code. The access code was not changed.
3206	MM Access Obj/SAT Busy	An administrator is making changes to the Meet-me Conference VDN, so the user cannot change the access code using a feature access code. Try again later.
3207	Merge Meet-me Conf call	A user tried to access an existing Meet-me Conference call and was denied.
3208	Serv Observ Meet-me VDN	A user tried to service observe a Meet-me Conference call. This is not allowed.
3209	Meet-me Conf call full	A user tried to access a Meet-me Conference call that was already full.
3210	Wrong MM Acc. code dialed	A user trying to access a Meet-me Conference call dialed the wrong access code.
3211	Chg Station no Cons/Perm	The station attempting to change the access code does not have console permissions COS.
3212	VDN not a meetme type	The VDN that was called is not a Meet-me Conference VDN.
3213	MM Invalid Conf Ctrlr Sta	If controlling extension is filled in and the station and controller do not match.

Summary of events (Continued)

Event type	Event description	Explanation
3214	MM Inv Trk not Remote Acc	The trunk used to access the Meet-me Conference is not a remote access trunk.
3215	MM Invalid Station Type	If controlling extension is blank and the station type is invalid (for example, and attendant console).
3216	Conf/Transfer 2 Meet-me	A user cannot conference or transfer another call into a Meet-me Conference call.
3217	MM Abbrev Dial Invalid	When changing a Meet-me Conference access code, the only entry that can be set up for abbreviated dialing is the feature-access-code (FAC). Any other entry generates the vector event.

Clearing events

When you have finished your review of the event log, you can remove events from the error log. You must be use superuser login ID to clear events.

To clear events from the error log, enter `clear events` at the command prompt and press **ENTER**. This command clears all events from the event buffer space within the error log. It does not delete any other entries in the error log.

Appendix E: Call Vectoring and BCMS interactions

Call Vectoring interacts with a management information system that helps to monitor and report on the activity within Call Vectoring. In most cases, the management system is the Basic Call Management System (BCMS).

BCMS collects and processes ACD information to generate various reports.

This appendix is intended to illustrate how this system interprets these management systems interpret and reports report on activity within Call Vectoring. Special emphasis is placed on interpreting and reporting on this activity as it occurs within splits during a series of Call Vectoring events.

Note:

Appendix A: Call Vectoring commands on page 153 provides a summary of the BCMS interactions with each Call Vectoring command (where applicable).

BCMS tracking in a Call Vectoring environment

Tracking is the identifying of call flows and other actions relevant to call handling. There are three classes of call flows: split flows, VDN flows, and vector flows. We are most concerned with tracking in the Call Vectoring environment. The specific types of call flows and actions in this environment that are tracked by the BCMS include the following:

- Inflows (flow ins)
- Outflows (flow outs)

The split supervisor can use VDN and vector flows to evaluate how effective vector programming is at the site in question. The supervisor can use split flows to determine the manner in which the splits at the site are handling incoming telephone calls.

Defining and interpreting call flows

The manner in which specific call flows are defined and interpreted depends upon the call flow class in question and the version of the switch being used.

The following sections define and interpret specific call flows according to these parameters.

VDN inflows and outflows

The following section discusses the specific VDN flows for BCMS.

BCMS standards

The following table illustrates how BCMS interprets specific VDN flows from the switch:

BCMS Standards for Interpreting VDN Flows	
Flow Type	**Interpretation**
VDN flow in	Not tracked.
VDN flow out	Calls that successfully flow out of a VDN to another VDN or to an external location via a **`route-to`** command.

Split inflows and outflows

The following sections discuss the various split flow types for the BCMS.

Before we detail how the BCMS interprets split flows, we should discuss the term primary split, since this concept plays a significant role in tracking. Primary split is defined as the first split in a VDN to which a call actually queues. Therefore, this split is not necessarily the first split referenced in the vector.

Another split becomes the primary split if either of the following events occurs:

- Call cannot queue to the originally-targeted split because the split has no queue slots available.
- Call leaves the VDN (via a **`route-to`** VDN command, for example) and is queued to another split as a result.

If the call leaves vector processing and does not queue to another split (as a result of a **`route-to extension`** command, for example), there is no new primary split.

With this discussion in mind, let’s take a look at the following table to see how BCMS interprets split flows for the switch:

BCMS Standards for Interpreting Split Flows

Flow Type	Interpretation
Inflow	Calls that ring at an agent in a split other than the primary split.
Outflow	Calls that are dequeued from a primary split via a **`route-to`** or **`messaging split`** command, or by ringing at or being answered by an agent in another split to which the call is also queued.

When a call is not answered (due to an outflow, abandon, busy, or disconnect), the call’s disposition is tracked for the primary split as long as the call is still queued when the call abandons, outflows, etc. However, if the call abandons or outflows from ringing, the disposition is recorded for the split for which it was ringing.

If the primary split in a VDN is unmeasured, an outflow, abandon, busy, or disconnect is not tracked for the call. Also, an answer is not tracked if the call is answered by an agent in the primary split.

Examples of split flow tracking

The following sections provide some examples of tracking in BCMS. Each section first presents a scenario of Call Vectoring events. The scenario is then followed by a table in which the tracking for the various splits involved is recorded. Following each “tracking table,” an explanation of the tracking procedure is provided.

The scenarios presented include the following:

- Call answered by a primary split.
- Call answered by a nonprimary split.
- Call abandoned from queue.
- Call answered by a primary split after a route to VDN.
- Call answered by a nonprimary split after a route to VDN.
- Call answered after a route to split.

Call answered by a primary split – The following scenario involves a call answered by the primary split. The scenario is as follows:

1. Call comes into a VDN whose vector queues the call to splits 1, 2 and 3.
2. Call is answered in split 1.

The following table shows the tracking table for this scenario:

Tracking for call answered by primary split

	Split tracking		
	1	**2**	**3**
BCMS	answer		

Comments:

- BCMS: No dequeue tracking item is available.

Call Answered by a non-primary split – The following scenario involves a call answered by a nonprimary split. The scenario is as follows:

1. Call comes into a VDN whose vector queues the call to splits 1, 2 and 3.
2. Call is answered in split 2.

The following table shows the tracking table for this scenario:

Tracking for call answered by non-primary split

	Split tracking		
	1	**2**	**3**
BCMS	outflow	inflow answer	

Comments:

- BCMS: Outflow is tracked in split 1 because the call is answered by an agent in another split to which the call is queued (that is, split 2). Inflow is tracked in split 2 because the call is answered in this split and the split is not the primary split. When the call is removed from split 3, outflow is not tracked in split 3 because this split is not the primary split.

Call Abandoned – The following scenario involves a call abandoned by the caller. The scenario is as follows:

1. Call comes into a VDN whose vector queues the call to splits 1, 2, 2 and 3.
2. Call is abandoned.

The following table shows the tracking table for this scenario:

Tracking for Abandoned Calls

	Split Tracking		
	1	**2**	**3**
BCMS	abandon		

Comments:

- BCMS: Abandon is tracked in split 1 because this split is the primary split. Tracking is not recorded in splits 2 and 3 because no dequeue tracking item is available.

Call answered by a primary split after a route to VDN – The following scenario involves a call answered by the primary split after a **`route-to`** VDN command is executed. The scenario is as follows:

1. Call comes into a VDN whose vector queues the call to splits 1, 2 and 3.
2. Vector executes a **`route-to VDN`** step.
3. Call is then queued to splits 4, 5 and 6.
4. Call is answered in split 4.

The following table shows the tracking table for this scenario:

Tracking for call answered by primary split after route to VDN

	Split tracking					
	1	**2**	**3**	**4**	**5**	**6**
BCMS	outflow			answer		

Comments:

Split 1 is the original primary split, because this is the first split to which the call actually queues. However, split 4 becomes the new primary split because:

- Call leaves the original VDN upon execution of the **`route-to VDN`** step.
- Split 4 is the first split to which the call queues upon execution of this step.
- BCMS: Outflow is tracked in split 1 because this split is the original primary split.

Call answered by the non-primary split after a route to VDN – The following scenario involves a call answered by the nonprimary split after a **`route-to VDN`** command is executed. The scenario is as follows:

1. Call comes into a VDN whose vector queues the call to splits 1, 2 and 3.
2. Vector executes a **`route-to VDN`** step.
3. Call is then queued to splits 4, 5 and 6.
4. Call is answered in split 5.

The following table shows the tracking table for this scenario:

Tracking for call answered by non-primary split after route to VDN

	Split tracking					
	1	**2**	**3**	**4**	**5**	**6**
BCMS	outflow			outflow	inflow answer	

Comments:

- BCMS: Outflow is tracked in split 1 because this split is the original primary split. Outflow is tracked in split 4 because this split becomes the new primary split after the **`route-to VDN`** step is executed. Finally, inflow is tracked in split 5 because the call is answered in this split, and the split is not the primary split.

Call answered after a route to split – The following scenario involves a call answered after it is routed to a split via a **`route-to digits`** or **`messaging split`** command. The scenario is as follows:

1. Call comes into a VDN whose vector queues the call to splits 1, 2 and 3.
2. Vector executes a **`route-to digits`** (or **`messaging split`**) step.
3. Call is queued to split 4 and answered by an agent in split 4.

The following table shows the tracking table for this scenario:

Tracking for call answered after route to split

	Split tracking			
	1	**2**	**3**	**4**
BCMS	outflow			answer

Comments:

- BCMS: Outflow is tracked in split 1 because this split is the original primary split, and the call is answered in split 4, which becomes the new primary split.

Evaluating split performance

By using the information presented to this point, along with the information from various reports (as discussed in the next section), the split supervisor can answer one or more questions concerning split performance and then make adjustments, if necessary. Here are some of the questions the supervisor can answer:

1. How many ACD calls offered to my split were “mine” (that is, were offered to this split as the primary split)?
2. How many of “my” ACD calls did “my” split not answer?
3. How many ACD calls that I didn’t answer weren’t “mine?”

The following sections present the answers to these questions from the perspective of the BCMS.

BCMS – The following answers reflect the use of BCMS:

- The number of calls offered to “my” split that were “mine” can be determined via examination of the BCMS Split Report. The algorithm is as follows: ACDCALLS + ABNCALLS + OUTFLOWCALLS - INFLOWCALLS (that is, the total number of calls answered plus the total number of calls abandoned from “my” split tagged as a primary split plus the number of calls that outflowed “my” split tagged as a primary split minus the number of calls answered that were not directed to “my” split tagged as a primary split).

Using BCMS reports to evaluate Call Vectoring activity

There are a number of BCMS reports that allow you to evaluate Call Vectoring activity. Some of these facets include the call flows present within Call Vectoring as well as the speeds at which calls are answered. The sections that follow identify and discuss the BCMS reports that indicate this activity.

BCMS reports

BCMS has a real-time split report, split historical reports, real-time VDN reports, and VDN historical reports. The following list identifies and describes several BCMS reports that summarize Call Vectoring activity. For more information on these and other related reports, refer to *Avaya MultiVantage Call Center Software Basic Call Management System (BCMS) Operations*, 555-230-706.

BCMS Split Report – Summarizes the call activity for an entire split. The information can be requested either daily or by the administered time period. Among other information, the report provides the total number of flow ins (inflows) and flow outs (outflows), the calls answered and calls abandoned. The report also provides the average speed of answer time for calls handled by the split during the indicated time period.

VDN Summary Report – Summarizes statistical information for all internally-measured VDNs. The information can be requested by the administered time interval or daily. The **`list bcms vdn`** report gives multiple time periods or days for a single VDN. The **`list bcms summary vdn`** report gives a one-line summary per vdn (with data from the specified times or days), but can give the data for numerous vdns.

The report also indicates the total number of flow outs, specifically, the number of calls that route to another VDN or to a destination external to the switch. However, calls that encounter a **`goto vector`** command are not shown as outflows. No further measurements are taken on the calls once the calls have outflowed. If an outflowed call later abandons, this is not indicated in the report.

Among other information, the VDN report provides a total for offered calls, answered calls, abandoned calls, and also one for calls that were either **`forced busy`** or **`forced disconnect`**.

VDN Real-Time Report – Provides statistical information including the number of calls currently waiting and the oldest call waiting. The VDN real-time report has the same characteristics as other real-time BCMS reports.

Appendix F: Operation details for the route-to command

The **`route-to`** command can be programmed with or without coverage. The following table summarizes the operation of the **`route-to`** command for each of the destination types and conditions associated with the commands.

Switch route-to command operation

Condition	cov = n Any Step	cov = y Any Step[1]
Invalid Destination[2]	Goes to next step, else stop	Goes to next step, else stop
VDN Extension[3]		
Vector Assigned	Goes to new vector	Goes to new vector
Vector Has No Steps	Stop[4]	Stop[4]
Station Extension Idle (all appearances idle)		
CF-ALL Active or CF-DA Applies	Forwards if possible, else next step, else stop[4]	Forwards if possible, else coverage, else busy
Coverage		
● DA Applies	Rings idle app.	Coverage on DA
● All Applies	Goes to next step, else stop[4]	Coverage
● SAC Applies	Rings idle appearance	Coverage
● None of Above Applies	Rings idle appearance	Call delivered and is allowed to cover

Switch route-to command operation (Continued)

Condition	cov = n Any Step	cov = y Any Step[1]
Station Extension Active (with idle 2-way app)		
CF-ALL Active	Forwards if possible, else next step, else stop[4]	Forwards if possible, else coverage, else busy
Coverage		
● DA Applies	Rings idle app (no DA timing)	Coverage on DA
● Ext Act Applies	Rings idle appearance	Coverage
● All Applies	Goes to next step, else stop[4]	Coverage
● SAC Applies	Rings idle appearance	Coverage
● None of Above Applies	Rings idle appearance	Call delivered and is allowed to cover
Station Extension Busy (no idle 2-way app)		
Extension in Hunt Grp (also see ACD Hunt Grp)	Queues if possible, else next step, else stop[4]	Queues if possible, else coverage, else busy
CF-ALL Active or CF-DA Applies	Forwards if possible, else next step, else stop[4]	Forwards if possible, else coverage, else busy
Call Waiting to Analog Sta Would Apply	Goes to next step, else stop[4]	Call waits
Coverage		
● Ext Act Applies	Goes to next step, else stop[4]	Coverage
● Ext Bsy Applies	Goes to next step, else stop[4]	Coverage
● All Applies	Goes to next step, else stop[4]	Coverage
● SAC Applies	Goes to next step, else stop[4]	Coverage
● None of Above Applies (or hunt, fwd, or cov destination is unavailable)	Goes to next step, else stop[4]	Busy tone given
Extension with Incompatible COR	Goes to next step, else stop.	Goes to next step, else stop.

Switch route-to command operation (Continued)

Condition	cov = n Any Step	cov = y Any Step[1]
Terminating Extension Group		
All Members Idle	Rings idle appearance	Call delivered and is allowed to cover
A Member Active on TEG	Goes to next step, else stop[4]	Coverage, else busy
No Idle App on Any Member	Goes to next step, else stop[4]	Coverage, else busy
Hunt Group Extension		
Idle Agent	Rings idle appearance	Call delivered and is allowed to cover
No Idle Agent		
● Call cannot queue	Goes to next step, else stop[4]	Busy tone given
● Call can queue	Call is queued	Call is queued
Extension on Another Node (Uniform Dialing Plan - UDP		
Trunk available	Call delivered	Call delivered
Trunk not available	Goes to next step, else stop[4]	Queues if possible, else reorder
Trunk Access Code (TAC) Destination		
Trk Grp No Dial Access	Goes to next step, else stop[4]	Routes to local attendant
Trunk Available	Call delivered	Call delivered
Trunk Not Available	Goes to next step, else stop[1]	Queues if possible, else reorder

Switch route-to command operation (Continued)

Condition	cov = n Any Step	cov = y Any Step[1]
AAR/ARS FAC Dest. (including Subnet Trkng)		
Trk Grp No Dial Access	Tries next route	Routes to local attendant
Trunk Available	Call delivered	Call delivered
Other Routes Avail	Call delivered	Tries next route
All Routes Busy		
• No Pattern Queuing	Goes to next step, else stop[4]	Reorder tone given
• Queuing Assigned	Goes to next step, else stop[4]	Queues to pattern
Attendant Queue (dial 0)		
Idle Atnd	Rings idle appearance	Call delivered and is allowed to cover
No Idle Atnd		
• Not In Night Svc	Call is queued	Call is queued
• In Night Svc		
— Dest. assigned	Delivered to night service	Delivered to night service
— Not assigned	Call is queued	Call is queued
Individual Attendant Access		
Attendant idle	Rings idle appearance	Call delivered and is allowed to cover
Attendant busy	Queues if possible else Goes to next step, else stop[4]	Queues if possible, else Busy tone given
CAS Attendant With Caller on Branch		
RLT available	Rings idle appearance	Call delivered and is allowed to cover
All RLTs busy	Queues if possible, else next step, else stop[4]	Queues if possible, else busy tone

Switch route-to command operation (Continued)

Condition	cov = n Any Step	cov = y Any Step[1]
Inter-PBX Atnd Calling		
Trunk Grp Controlled	Routes to local atendant	Routes to local atendant
Trunk Available	Call delivered	Call delivered
Trunk Not Available	Goes to next step, else stop[4]	Reorder tone given
Look Ahead Interflow (LAI) (feature active & routes over ISDN-PRI facility)[5]		
B-Channel Not Available	Goes to next step, else stop[4]	Queues if possible, else reorder
B-Channel Available and Receiving Switch:		
● Accepts	Interflow succeeds[6]	Call cut-through
● Rejects	Goes to next step, else stop[4]	Call gets busy/disconnect
Receiving Switch w LAI Acting as Tandem Sees from Remote Receiving Switch:		
● Call Rejected	Goes to next step at receiving switch, else sending switch considers call rejected after 2-minute timeout	Call gets busy/disconnect
● **`if interflow-qpos`**	Determines if queued call is eligible for interflow	Determines if queued call is eligible for interflow

[1] When the **`with coverage`** option is set to **y**, the call is removed from vector processing when the route-to step is reached, regardless of facility or remote switch availability. The call is taken out of any split queue, and any feedback, such as music or ringback, is removed, even if the destination is not available. If the call is subsequently "rejected" by the receiving switch vector, subsequent call treatment is defined by the "rejection" command (either busy or forced disconnect). The call is treated as though the destination is directly dialed (see footnote 3 for related information). This includes coverage, forwarding, treatments for calls that cannot be completed (busy reorder, and intercept) and displays. The answering station sees only caller name and number. A call routed via an **`adjunct routing`** command is treated the same way as a call that is routed via a **`route-to with coverage y`** command.

2. Invalid destinations include the following: empty (for example, zero collected digits) or invalid route-to destination number, unassigned extension number, incomplete number of digits for AAR/ARS pattern, non-AAR/ARS feature access code (FAC), maintenance busy station extension, COR of the VDN that prevents access (for example, origination restricted), FRL of a VDN that is lower than required for the AAR/ARS pattern access, no routes assigned to the AAR/ARS pattern, incompatible calling and destination partitions, ACTGA trunk group destination, or an off-net forwarding destination. If a TAC (trunk access code) destination is involved, and if the TAC is for a CO/FX trunk with a **`route-to with coverage n`** step, the digits entered must match a valid ARS analysis string. If not, the destination is considered invalid. For other trunk types with a **`route-to number`** or **`route-to digits with coverage n`** step, the step succeeds when the trunk is seized (that is, vector processing stops). For a **`route-to with coverage y`** step, the step succeeds if the TAC is assigned.

3. A call that routes to a VDN via the **`route-to number with cov = y unconditionally`** command behaves like a directly- dialed call instead of a VDN call. Therefore, the terminating station's display only shows the originating station information and does not show the VDN information (for other types of VDN calls, the terminating station would see the VDN name).

4. The interaction "Stop" means the following: vector processing is stopped, the call remains queued to a split, and the caller continues to hear feedback initiated by a previous step. In the case where the **`route-to`** command fails and processing stops (due to a busy station or trunk group destination), retry can be implemented in the vector. Retrying is accomplished by including an unconditional **`goto`** step as the last step to allow for a loop back to the **`route-to`** command. Use of an intermediate **`wait-time`** command step with appropriate feedback and delay interval is strongly recommended in order to reduce processor occupancy.

5. With one exception, any **`route-to with cov= y`** step that routes over ISDN-PRI facilities cancels Look-Ahead Interflow. The exception occurs when a call reaches a vector via coverage to a VDN. Calls that cover to a VDN will not be further forwarded or otherwise redirected. For covered calls, a **`route-to`** command with coverage set to `y` functions as though coverage were set to `n`. Thus, a **`route-to with coverage y`** will route covered calls via LAI over ISDN facilities if LAI is enabled.

6. On the sending switch, the call is removed from vector processing (that is, the call is taken out of any split queue and any feedback, such as music or ringback, is removed).

Appendix G: Setting up a contact center

Contact center managers need some key indicators to measure ACD performance at their site. Usually, in setting up a contact center, several factors involving call management are considered. The following list identifies and defines the most common of these factors, and it provides a typical question that might be asked. In addition, an insurance company example will be used to discuss the different options in this appendix.

Volume – Number of calls going in or out of the ACD. (How many calls did Split 1 answer?)

Productivity – Call volume per unit of time. (How many calls did Split 1 answer between 8 a.m. and 9 a.m.?)

Utilization – Overall use of the phone center. (What was my agent occupancy?)

Accessibility**: –** Availability of lines and agents when customers call the ACD. (Were lines busy when customers called or did they have to wait too long?)

Quality of Service**: –** Accuracy of information, a pleasant manner, responsiveness to caller concerns, successful completion of business, and efficient time utilization. (Was the caller given good service?)

This chapter explains how to set up a contact center for customers with Call Vectoring.

Call Vectoring option

To set up a contact center that has Call Vectoring, do the following:

1. Determine your contact center's objectives. Think about how you want your contact center to handle calls and also about what you want your contact center to achieve.

   A company's basic goals are to increase profits and market share and to decrease costs. The purpose of setting up a contact center is to monitor these goals using the BCMS reports. It is best to have more than one objective. (Some customers set and then live by only one objective.) Contact center objectives must then be created to meet the goals. These objectives must be communicated to the Split Supervisor or to the Administrator managing the contact center.

   The following list provides an example set of contact center objectives:

   - Establish the following measured entities:
     - Average Speed of Answer = 15 seconds
     - Abandon Rate <= 3%
     - Average Talk Time = 2 1/2 minutes
     - ACD calls per agent = 80 to 90 per day
     - Number of calls in queue = 6
     - Percentage of calls answered within the service level = 95%
     - Agent occupancy > 90%
     - Percentage of trunks busy < 3%
   - Generate revenue through the contact center.
   - Train agents to back up each other.
   - Adequately train agents to provide service that meets customer expectations.

2. Review your existing operation and determine your contact center needs (see Worksheet #2: Current split operation on page 239 and Customer/contact center needs guidelines on page 235).

3. On the switch, assign a unique Hunt Group number and Call Distribution method to each caller need. This number will be your split number (see Non-EAS Worksheet #3: Customer needs on page 240 and Customer/contact center needs guidelines on page 235).

4. Assign DNIS (Dialed Number Identification Service) (that is, the number dialed) as a Vector Directory Number (VDN) (see Customer/contact center needs guidelines on page 235).

   As an option, you can assign one VDN for a main number and use Call Prompting to route the call to the proper split.

   The following table illustrates the guidelines given up to this point.

Customer/contact center needs guidelines

Need	Split number (hunt group)	Call distribution[1]	VDN
New policy	1	UCD	555-6543
Questions about policy, Rate Quotes, Billing	2	UCD	555-6432
Spanish speaking for policy, service, and claims	3	DDC	555-6321
Claims	4	UCD	555-6210

[1] Options include Direct Department Calling (DDC) and Uniform Call Distribution (UCD).

   Notice that this contact center has only one split for all Spanish calls. However, resources permitting, you could create a New Policy split, a Service split, and a Claims split, each containing agents who speak Spanish. As an alternative, you could use one main VDN to point to a Call Prompting vector designed to route the calls to the splits.

5. On the switch, assign extensions to the agents' physical terminal locations.

6. On the switch, assign agent extensions to splits. More than four splits can be assigned to an agent; however, the agent can log into a maximum of four splits.

   The following table illustrates the assignment of agent extensions to splits:

Agent extension/split assignments

Split (hunt group)	Agent extensions
1 - Sales	1231, 1232, 1233, 1234, 1235, 1236, 1237, 1238, 1239
2 - Service	1231, 1232, 1234, 1238, 1239, 1240
3 - Spanish	1238, 1240, 1245
4 - Claims	1238, 1239, 1240, 1241, 1242

7. On the switch, assign a vector to each VDN. A VDN can point to only one vector. However, a vector can have more than one VDN pointing to it.

   The following table illustrates VDN/vector assignments.

VDN/vector assignments

VDN	Vector
6543	1 (Sales)
6432	2 (Service)
6321	3 (Spanish)
6210	4 (Claims)

8. On the switch, write your vectors.

   Your vectors should match your contact center objectives. To meet these objectives, you must make a number of relevant decisions (for example, you may decide how soon you want to enlarge an agent pool or what kind of treatment the caller should receive). If your VDN and vector reports do not satisfy your contact center objectives, you must consider your alternatives (for example, you may deem it necessary to train agents or to increase the amount of time elapsed from when a call queues to one split and then to another split).

   The following lists indicate the actions produced by two different vectors:

   Actions Produced by Vector #1:

   a. Tell the caller to select one of the following prompts:
      - 1 = Sales
      - 2 = Service
      - 3 = Spanish
      - 4 = Claims
      - Nothing or 0 = Service

   b. Queue the call.

   c. Provide an announcement to the caller.

9. Actions Produced by Vector #2:

   a. Queue the call to the correct service at a medium priority.

   b. If no agents are available, provide a message and then play music.

   c. If the call is not answered within 10 seconds, provide a second message and then play music.

d. If the call is not answered within 7 more seconds, queue the call to the Service split.

e. If the call is not answered within 7 more seconds, queue the call to the Spanish split at a high priority.

Note:

> A **`check split`** command queues the call to up to three splits if the conditions are met. If the conditions are not met, the **`check split`** command may not get read again (if the vector step in which it appears is not executed again).

10. Once your system is up and operational, you will need to monitor it to ensure you are meeting your contact center objectives. The call management system can be used to monitor many of your objectives. Some objectives will need to be monitored and have adjustments made in real time. For example, if the number of calls waiting, average speed of answer, or percent answered within a service level is not meeting your objectives, you might want to immediately move some agents, direct calls to another vector, or look-ahead interflow some calls. Other items such as agent occupancy and percent all trunks busy may only need to be monitored daily to look for trends.

Worksheet #1: Contact center objectives

What are my contact center objectives?

Worksheet #2: Current split operation

Split ____________________

Primary Backup __________________	**Secondary Backup** _________________	**Tertiary Backup** __________________

List your customer/caller needs and your agent knowledge levels for this split.

1. ____

2. ____

3. ____

4. ____

5. ____

6. ____

Split ____________________

Primary Backup __________________	**Secondary Backup** _________________	**Tertiary Backup** __________________

List your customer/caller needs and your agent knowledge levels for this split.

1. ____

2. ____

3. ____

4. ____

5. ____

6. ____

Non-EAS Worksheet #3: Customer needs

Contact center needs	Split number (hunt group)	Call distribution	VDN

Worksheet #4: Vector design

Vector #	Name:	Description:
Assigned VDNs:		
Assigned Trunk Groups:		
1.		
2.		
3.		
4.		
5.		
6		
7.		
8.		
9.		
10.		
11.		
12.		
13.		
14.		
15.		
16.		
17.		
18.		
19.		

20.

21.

22.

23.

24.

25.

26.

27.

28.

29.

30.

31.

32.

Appendix H: Improving performance

This appendix provides recommendations on how to write vectors that promote favorable performance practices. Improved performace depends on the following basic principles:

- Minimize the number of vector steps to process a call.
- Avoid vector steps which have a substantial probability of failure, such as:
  - Calls made outside of business hours
  - Queues to groups with less than desirable resources or characteristics.

The most wasteful use of processing resources is frequently caused by inefficient looping. For example, performance could be compromised when a vector loops through steps too often. This is especially true with long queue times.

Some examples with looping are discussed and recommendations are given on how to maximize performance. They are:

- Audible Feedback
- Check
- Examples other than looping, such as after business hours, are also discussed.

All looping examples in this appendix use only loops within a single vector. It is important to also 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:

Remember to test vectors for performance in addition to call flow.

Looping examples

Audible feedback

Recommendation: Evaluate the length of the wait period between repetitions of an announcement and increase the length, if possible. For optimum performance, add a second announcement after the initial announcement and repeat the second announcement less often.

The first example repeats the “All representative are busy. Please hold.” 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 next example repeats the announcement only every 60 seconds, thus 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
```

The next example adds a second announcement, “All representatives are still busy. Please hold.” in addition to the initial announcement and repeats the second announcement less often (every 120 seconds), thus improving performance again.

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 while processing the call. Assumption is that the first announcement is 3 seconds long and the second announcement is 4 seconds long.

Approximate number of vector steps executed for the audible feedback examples

Initial conditions	**Example:** **10-second announcement interval**	**Example:** **60-second announcement interval**	**Example:** **Follow-up announcement**
An agent is available in split 1	1	1	1
Queueing time of 5 minutes	70	15	9

When a call is queued for 5 minutes, the number of vector steps drops dramatically when the amount of time between announcements is increased, and drops even more when a second announcement is added, and the amount of 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.

Check

Recommendation: When using check commands to queue a call to backup splits, ensure that an adequate amount of time has elapsed before checking the backup splits again.

The next example checks backup splits continuously as long as the call is in queue.

Example: Continuous check

```
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
```

The next example adds a delay of 10 seconds to ensure that some time has elapsed before checking the backup splits again.

Example: Check with 10 second delay

```
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
```

Since the agent availability status may not be likely to change every 10 seconds, it may make sense to increase the wait time to 30 seconds, as shown in the example in The following example.

Example: Check with 30 second delay

```
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.

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
Queueing time of 5 minutes	up to 1,000	190	65

When a call is queued for 5 minutes, the number of vector steps drops dramatically when a delay is added before checking the backup splits again, and drops even more 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.

Other examples

After business hours

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 the call is made 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 is made 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 is queued after business hours and the call is terminated only when the caller hangs up. As shown in the second example, it is more economical to test for business hours before queuing a call.

Glossary

AAR
See Automatic Alternate Routing (AAR).

abandoned call
An incoming call in which the caller hangs up before the call is answered.

ACD
See Automatic Call Distribution (ACD).

ACD agent
See agent.

ACD
See Automatic Call Distribution (ACD). ACD also refers to a work state in which an agent is on an ACD call.

ACD work mode
See work mode.

active-notification association
A link that is initiated by an adjunct, allowing it to receive event reports for a specific switch entity, such as an outgoing call.

active-notification call
A call for which event reports are sent over an active-notification association (communication channel) to the adjunct. Sometimes referred to as a monitored call.

active notification domain
VDN or ACD split extension for which event notification has been requested.

after-call work (ACW) mode
A mode in which agents are unavailable to receive ACD calls. Agents enter the ACW mode to perform ACD-related activities such as filling out a form after an ACD call.

agent
A person who receives calls directed to a split. A member of an ACD hunt group or ACD split. Also called an ACD agent.

agent report
A report that provides historical traffic information for internally measured agents.

agent selection method
The method the switch uses to select an agent in a hunt group when more than one agent is available to receive the next call: UCD-MIA, UCD-LOA, EAD-MIA, or EAD-LOA

ANI
See Automatic Number Identification (ANI).

ARS
See Automatic Route Selection (ARS).

auto-in trunk group Trunk group for which the CO processes all of the digits for an incoming call. When a CO seizes a trunk from an auto-in trunk group, the switch automatically connects the trunk to the destination — typically an ACD split where, if no agents are available, the call goes into a queue in which callers are answered in the order in which they arrive.

Auto-In Work mode One of four agent work modes: the mode in which an agent is ready to process another call as soon as the current call is completed.

Automatic Alternate Routing (AAR) A feature that routes calls to other than the first-choice route when facilities are unavailable.

Automatic Call Distribution (ACD) A feature that answers calls, and then, depending on administered instructions, delivers messages appropriate for the caller and routes the call to an agent when one becomes available.

Automatic Call Distribution (ACD) split A method of routing calls of a similar type among agents in a contact center. Also, a group of extensions that are staffed by agents trained to handle a certain type of incoming call.

Automatic Number Identification (ANI) Representation of the calling number, for display or for further use to access information about the caller. Available with Signaling System 7.

Automatic Route Selection (ARS) A feature that allows the system to automatically choose the least-cost way to send a toll call.

Aux-Work mode A work mode in which agents are unavailable to receive ACD calls. Agents enter Aux-Work mode when involved in non-ACD activities such as taking a break, going to lunch, or placing an outgoing call.

available agent strategy Part of the Best Service Routing feature, the available agent strategy determines how BSR commands in a vector identify the best split or skill when several have available agents. The possible available agent strategies are: UCD-MIA, UCD-LOA, EAD-MIA, and EAD-LOA.

BCMS Basic Call Management System

CACR Cancellation of Authorization Code Request

call vector A set of up to 15 vector commands to be performed for an incoming or internal call.

cause value A value is returned in response to requests or in event reports when a denial or unexpected condition occurs. ASAI cause values fall into two coding standards: Coding Standard 0 includes any cause values that are part of AT&T and CCITT ISDN specifications; Coding standard 3 includes any other ASAI cause values. This document uses a notation for cause value where the coding standard for the cause is given first, then a slash, then the cause value. Example: CS0/100 is coding standard 0, cause value 100.

Class of Restriction (COR) A feature that allows up to 64 classes of call-origination and call-termination restrictions for telephone, telephone groups, data modules, and trunk groups. See also Class of Service (COS).

Class of Service (COS) A feature that uses a number to specify if voice-terminal users can activate the Automatic Callback, Call Forwarding All Calls, Data Privacy, or Priority Calling features. See also Class of Restriction (COR).

COR See Class of Restriction (COR).

COS See Class of Service (COS).

coverage answer group A group of up to eight telephones that ring simultaneously when a call is redirected to it by Call Coverage. Any one of the group can answer the call.

coverage call A call that is automatically redirected from the called party's extension to an alternate answering position when certain coverage criteria are met.

coverage path The order in which calls are redirected to alternate answering positions.

coverage point An extension or attendant group, VDN, or ACD split designated as an alternate answering position in a coverage path.

covering user A person at a coverage point who answers a redirected call.

DAC 1. Dial access code or Direct Agent Calling

DDC Direct Department Calling

DIVA Data In/Voice Answer

DNIS Dialed-Number Identification Service

domain VDNs, ACD splits, and stations. The VDN domain is used for active-notification associations. The ACD-split domain is for active-notification associations and domain-control associations. The station domain is used for the domain-control associations.

ETN Electronic tandem network

extension-in Extension-In (ExtIn) is the work state agents go into when they answer (receive) a non-ACD call. If the agent is in Manual-In or Auto-In and receives an extension-in call, it is recorded by CMS as an AUX-In call.

extension-out The work state that agents go into when they place (originate) a non-ACD call.

extension A number by which calls are routed through a communications system or, with a Uniform Dial Plan (UDP) or main-satellite dialing plan, through a private network.

FAC Feature Access Code

hunt group A group of extensions that are assigned the Station Hunting feature so that a call to a busy extension reroutes to an idle extension in the group. See also ACD work mode.

ICDOS International Customer-Dialed Operator Service

ICHT Incoming call-handling table

ICI Incoming call identifier

interflow To route an incoming call to an external switch without answering it at the origin switch.

intraflow The ability for calls to redirect to other splits on the same PBX on a conditional or unconditional basis using call coverage busy, don't answer, or all criteria.

internal measurements BCMS measurements that are made by the system.

LDN Listed directory number

Manual-In work mode One of four agent work modes: the mode in which an agent is ready to process another call manually. See Auto-In Work mode for a contrast.

message center An answering service that supplies agents to and stores messages for later retrieval.

message center agent A member of a message-center hunt group who takes and retrieves messages for voice-terminal users.

monitored call See active-notification call.

other split The work state that indicates that an agent is currently active on another split's call, or in ACW for another split.

redirection criteria Information administered for the coverage path of each telephone, that determines when an incoming call is redirected to coverage.

Redirection on No Answer

An optional feature that redirects an unanswered ringing ACD call after an administered number of rings. The call is then redirected back to the agent.

report scheduler

Software that is used in conjunction with the system printer to schedule the days of the week and time of day that the desired reports are to be printed.

split

See ACD work mode.

split condition

A condition whereby a caller is temporarily separated from a connection with an attendant. A split condition automatically occurs when the attendant, active on a call, presses the start button.

split number

The split’s identity to the switch and BCMS.

split report

A report that provides historical traffic information for internally measured splits.

split (agent) status report

A report that provides real-time status and measurement data for internally measured agents and the split to which they are assigned.

staffed

Indicates that an agent position is logged in. A staffed agent functions in one of four work modes: Auto-In, Manual-In, ACW, or AUX-Work.

system report

A report that provides historical traffic information for internally measured splits.

system-status report

A report that provides real-time status information for internally measured splits.

to control

An application can invoke Third Party Call Control capabilities using either an adjunct-control or domain-control association.

to monitor

An application can receive event reports on an active-notification, adjunct-control, or domain-control association.

UCD

Uniform call distribution

VDN

See vector directory number (VDN).

vector directory number (VDN)

An extension that provides access to the Vectoring feature on the switch. Vectoring allows a customer to specify the treatment of incoming calls based on the dialed number.

vector-controlled split A hunt group or ACD split administered with the vector field enabled. Access to such a split is possible only by dialing a VDN extension.

work mode One of four states (Auto-In, Manual-In, ACW, AUX-Work) that an ACD agent can be in. Upon logging in, an agent enters AUX-Work mode. To become available to receive ACD calls, the agent enters Auto-In or Manual-In mode. To do work associated with a completed ACD call, an agent enters ACW mode.

work state An ACD agent may be a member of up to three different splits. Each ACD agent continuously exhibits a work state for every split of which it is a member. Valid work states are Avail, Unstaffed, AUX-Work, ACW, ACD (answering an ACD call), ExtIn, ExtOut, and OtherSpl. An agent’s work state for a particular split may change for a variety of reasons (example: when a call is answered or abandoned, or the agent changes work modes). The BCMS feature monitors work states and uses this information to provide BCMS reports.

Index