Configuring Modem Support and Other Asynchronous Features

This chapter describes how to allow asynchronous connections, such as analog modem calls, to enter an access server. It also describes how to set up advanced asynchronous features, such as asynchronous dynamic routing and chat scripts.

For a complete description of the modem support commands in this chapter, refer to the "Modem Support and other Asynchronous Commands" chapter of the *Dial Solutions Command Reference*. To locate documentation of other commands that appear in this chapter, use the command reference master index or search online.

The following sections are provided in this chapter:

- Basic Concepts
- Basic Modem Configuration Task List
- Monitor and Maintain Connections
- Troubleshoot Externally Attached Modems
- Set Up Advanced Features
- Use Chat Scripts
- Configuration Examples

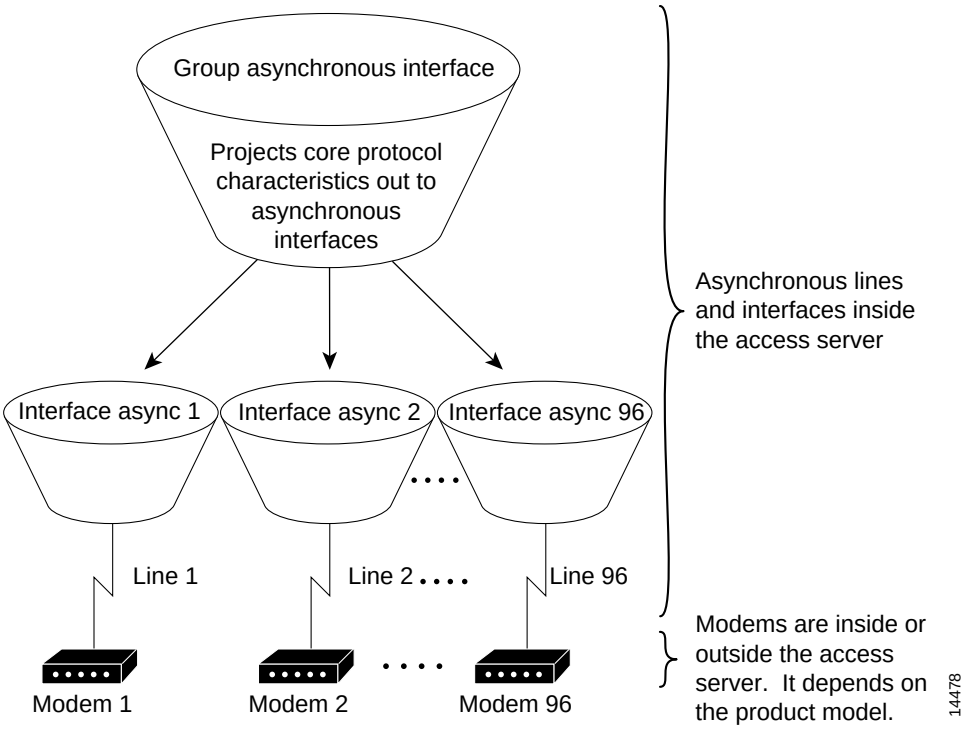
Basic Concepts

Modems attach to asynchronous lines, which in turn attach to asynchronous interfaces. Depending on the type of access server you have, these components appear outside or inside the physical chassis. Figure 15 shows the logical relationship between modems, asynchronous lines, asynchronous interfaces, and group asynchronous interfaces. All these components work together to deliver packets as follows:

- Asynchronous calls come into the modems from the POTS or PSTN network.
- Modems pass packets up through asynchronous lines.
- Asynchronous interfaces clone their configuration information from group asynchronous interfaces.

Note The number of interfaces and modems varies between access server product models.

Figure 15 Modems, Lines, and Asynchronous Interfaces



Related Hardware Differences

Deciding which asynchronous features to use, to some degree, depends on your hardware configuration. All Cisco access servers must have their asynchronous interfaces and lines configured for network protocol support. Commands entered in asynchronous interface mode configure protocol-specific parameters for asynchronous interfaces, whereas commands entered in line configuration mode configure the physical and logical aspects for the same port.

Modems inside high-end access servers need localized modem country code. This code is projected from the Cisco IOS software to the onboard modems using the **modem country** {**mica** | **microcom_hdms**} *country* command. The following are high-end access servers: Cisco AS5800, Cisco AccessPath, Cisco AS5300, and the Cisco AS5200.

Modems externally attached to low-end access servers need to receive initialization strings from the **modem autoconfigure discovery** command. For troubleshooting tips, see the section "Troubleshoot Externally Attached Modems." The following are low-end access servers: Cisco AS2511-RJ, Cisco AS2509-RJ, Cisco 2509, Cisco 2511, and the Cisco 2512.

Figure 16 shows a Cisco AS2511-RJ access server. Figure 17 shows a Cisco AS5300 access server. Notice that modems are either inside or outside the chassis, depending on the product model.

Figure 16 Cisco AS2511-RJ Access Server

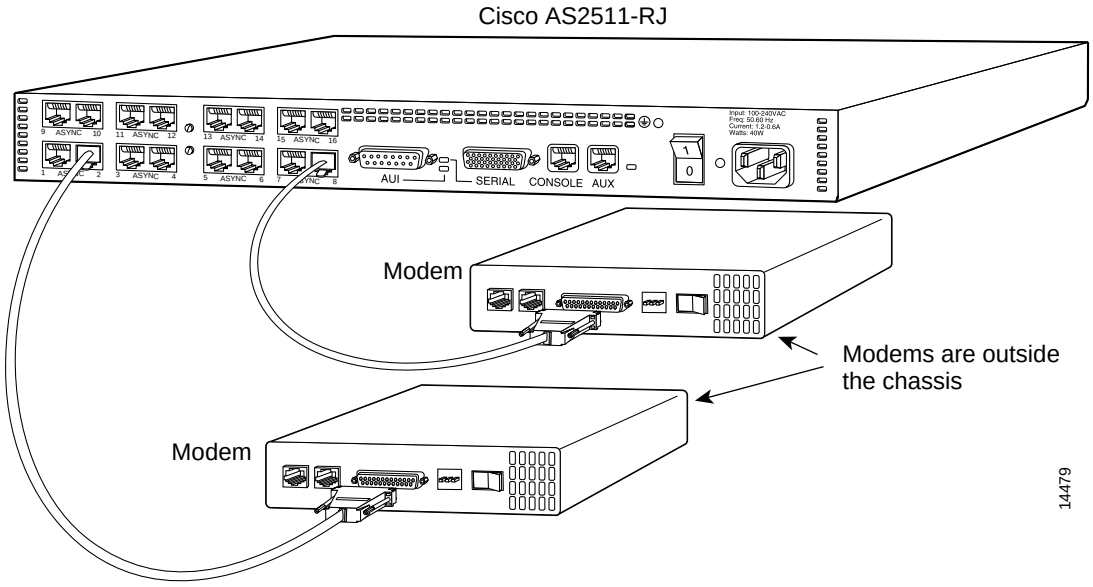
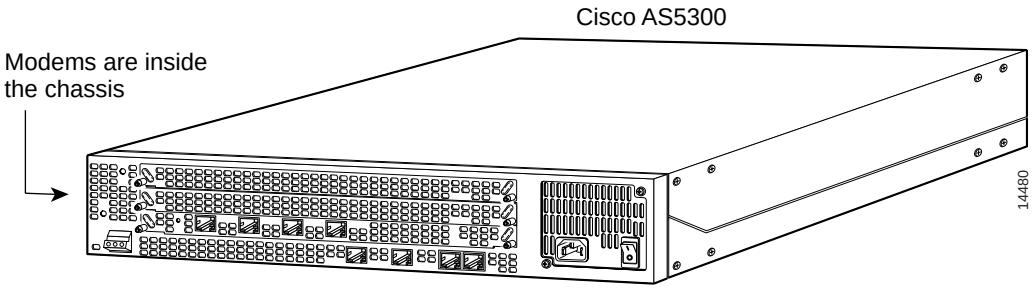


Figure 17 Cisco AS5300 Access Server



Asynchronous Modem Lines

Asynchronous line configuration commands configure ports for the following options:

- Physical layer options (such as modem configuration)
- Security for login
- ARA protocol configuration
- Autoselect to detect incoming protocols (ARA and PPP)

To enter line configuration mode, first connect to the console port of the access server and enter privileged EXEC mode. Then enter global configuration mode and finally enter line configuration mode for the asynchronous lines that you want to configure.

The following example is for a Cisco AS5800 access server, which is used as a high-density dial-in solution.

```
as5800# configure terminal
as5800(config)# line 1/2/00 1/9/71
as5800(config-line)# session-timeout 30
as5800(config-line)# exec-timeout 30 0
as5800(config-line)# absolute-timeout 240
as5800(config-line)# autoselect during-login
as5800(config-line)# autoselect ppp
as5800(config-line)# modem InOut
as5800(config-line)# transport preferred none
as5800(config-line)# transport input all
```

The following is an example of one asynchronous line configuration on a Cisco AS2511-RJ access server that is used in an asynchronous backup DDR scenario:

```
as2511> enable
as2511# configure terminal
as2511(config)# line 1
as2511(config-line)# modem InOut
as2511(config-line)# speed 115200
as2511(config-line)# transport input all
as2511(config-line)# flowcontrol hardware
```

Asynchronous Interfaces

Interfaces enable the Cisco IOS software to use routing functions. Specifically, you configure asynchronous interfaces to support PPP connections. Asynchronous interfaces are configured on access servers for the following functions:

- Network protocol support (such as IP, IPX, or AppleTalk)
- Encapsulation support (such as PPP)
- IP client addressing options (default and/or dynamic)
- IPX network addressing options
- PPP authentication

The following is an example of one asynchronous interface configuration on a Cisco AS2511-RJ access server that is used in an asynchronous backup DDR scenario:

```
as2511(config)# interface async 1
as2511(config-if)# description ASYNC LINE 5293731 TO HIGHWAY
as2511(config-if)# encapsulation ppp
as2511(config-if)# async default routing
as2511(config-if)# async mode dedicated
as2511(config-if)# dialer in-band
as2511(config-if)# dialer map ip 192.168.10.2 name Router2 broadcast
as2511(config-if)# dialer-group 1
as2511(config-if)# ppp authentication chap
```

Group Asynchronous Interfaces

To configure multiple asynchronous interfaces at the same time (with the same parameters), you can assign each asynchronous interface to a group and then configure the group. Configurations throughout this guide configure group asynchronous interfaces, rather than configuring each interface separately.

Note After assigning asynchronous interfaces to a group, you cannot configure these interfaces separately. If you want to configure different attributes on different asynchronous interfaces, do not assign them to the group or assign different interfaces to different groups. For example, on a Cisco AS5300 access server in a T1 configuration, you could assign asynchronous interfaces 1 to 48 as part of one group (such as group-async1) and asynchronous interfaces 49 to 96 as part of another group (group-async2). You can also use the **member** command to perform a similar grouping function.

To configure a group asynchronous interface, specify the group async number (an arbitrary number) and the group range (beginning and ending asynchronous interface number). The following example shows the process of creating and configuring a group asynchronous interface for asynchronous interfaces 1 through 96 on a Cisco AS5300 access server, which is loaded with 96 56K MICA modems:

```
as5300(config)# interface group-async 1
as5300(config-if)# ip unnumbered ethernet 0
as5300(config-if)# encapsulation ppp
as5300(config-if)# async mode interactive
as5300(config-if)# ppp authentication chap pap
as5300(config-if)# peer default ip address pool default
as5300(config-if)# group-range 1 96

Building configuration...

as5300(config-if)#
```

The following example is for a Cisco AS5800 access server, which is used as a high-density dial-in solution.

```
as5800(config)# interface group-async 0
as5800(config-if)# ip unnumbered FastEthernet0/2/0
as5800(config-if)# encapsulation ppp
as5800(config-if)# async mode interactive
as5800(config-if)# peer default ip address pool default
as5800(config-if)# no cdp enable
as5800(config-if)# ppp authentication chap
as5800(config-if)# hold-queue 10 in
as5800(config-if)# group-range 1/2/00 1/9/71

Building configuration...

as5800(config-if)#
```

Line and Modem Numbering Issues

The TTY line numbering scheme used by your access server or router is specific to your product and its hardware configuration. Refer to the product-specific documentation that came with your product for line numbering scheme information.

For example, the Cisco AS5200 access server has TTY lines that map directly to integrated modems as shown in Table 4. Depending on the shelf, slot, port physical architecture of the access server, the modem and TTY line number schemes will change.

As shown in Table 4, TTY lines 1 through 24 directly connect to modems 1/0 through 1/23, which are installed in the first chassis slot in this example. The TTY lines 25 through 48 directly connect to modems 2/0 through 2/23, which are installed in the second slot.

Table 4 TTY Lines Associated to Cisco AS5200 Modems

TTY Line	Slot/ Modem Number	TTY Line	Slot/ Modem Number
1	1/0	25	2/0
2	1/1	26	2/1
3	1/2	27	2/2
4	1/3	28	2/3

Table 4 TTY Lines Associated to Cisco AS5200 Modems (continued)

TTY Line	Slot/ Modem Number	TTY Line	Slot/ Modem Number
5	1/4	29	2/4
6	1/5	30	2/5
7	1/6	31	2/6
8	1/7	32	2/7
9	1/8	33	2/8
10	1/9	34	2/9
11	1/10	35	2/10
12	1/11	36	2/11
13	1/12	37	2/12
14	1/13	38	2/13
15	1/14	39	2/14
16	1/15	40	2/15
17	1/16	41	2/16
18	1/17	42	2/17
19	1/18	43	2/18
20	1/19	44	2/19
21	1/20	45	2/20
22	1/21	46	2/21
23	1/22	47	2/22
24	1/23	48	2/23

Basic Modem Configuration Task List

Perform the following required tasks to configure modem support for high-end access servers, such as the Cisco AS5800, AS5300, and AS5200:

- Configure the Modems and Lines
- Create the Group Asynchronous Interface

Configure the Modems and Lines

You must configure the modem lines and set the country code to enable asynchronous connections into your access server.

To configure the modems and line, use the following commands beginning in global configuration mode:

Step	Command	Purpose
1	**modem country mica** *country* or **modem country microcom_hdms** *country*	Depending on the type of modems loaded in your access server, specify the modem vendor and country code.[1] This step is only for high-end access servers, such as the Cisco AS5300.
2	**line** *line-number ending-line-number*	Enter the number of modem lines to configure. Usually this range is equal to the number of modems in the access server.[2]
3	**transport** {**input** | **output**} **all**	Allow all protocols to be used when connecting to the line. For outgoing calls, choose the **output** option. For incoming calls, use **input**.
4	**autoselect** {**arap** | **ppp** | **slip**}	Configures the line to automatically startup an ARA, PPP, or SLIP session.
5	**autoselect during login**	Configure the lines to autodetect the connection oriented protocol and display the username:password prompt upon login.
6	**login authentication dialin**	Enable authentication across all asynchronous modem logins.
7	**modem dialin**	Enable incoming calls.[3]
8	**exit**	Return to global configuration mode.

1 For a comprehensive list of modem country codes, see the **modem country mica** command and the **modem country microcom_hdms** command in the *Dial Solutions Command Reference*. For lower-end access servers, use the **modem autoconfigure discovery** command.

2 If you create more than one group asynchronous interfaces then assign different ranges of modems to each group interface. One modem cannot be used in more than one group asynchronous statement.

3 To enable incoming and outgoing calls, enter the **modem inout** command instead.

Verify the Modem Configuration

To verify your modem configuration:

- Enter the **show line** command to display a summary for all the lines:

```
NAS# show line
 Tty Typ     Tx/Rx     A Modem  Roty AccO AccI  Uses    Noise   Overruns
*  0 CTY               -    -      -    -    -     0        0        0/0
I  1 TTY 115200/115200 - inout     -    -    -     0        0        0/0
I  2 TTY 115200/115200 - inout     -    -    -     0        0        0/0
   3 TTY 115200/115200 - inout     -    -    -     0        0        0/0
   4 TTY 115200/115200 - inout     -    -    -     0        0        0/0
   5 TTY 115200/115200 - inout     -    -    -     0        0        0/0
   6 TTY 115200/115200 - inout     -    -    -     0        0        0/0
   7 TTY 115200/115200 - inout     -    -    -     0        0        0/0
   8 TTY 115200/115200 - inout     -    -    -     0        0        0/0
   9 TTY 115200/115200 - inout     -    -    -     0        0        0/0
  10 TTY 115200/115200 - inout     -    -    -     0        0        0/0
.
.
.
  90 VTY               -    -      -    -    -     0        0        0/0

NAS# show line 1
 Tty Typ     Tx/Rx     A Modem  Roty AccO AccI  Uses    Noise   Overruns
I  1 TTY 115200/115200 - inout     -    -    -     0        0        0/0

Line 1, Location: "", Type: ""
Length: 24 lines, Width: 80 columns
Baud rate (TX/RX) is 115200/115200, no parity, 1 stopbits, 8 databits
Status: none
Capabilities: Hardware Flowcontrol In, Hardware Flowcontrol Out
  Modem Callout, Modem RI is CD, Line usable as async interface
Modem state: Idle
Special Chars: Escape  Hold  Stop  Start  Disconnect  Activation
                ^^x    none   -     -       none
Timeouts:      Idle EXEC    Idle Session   Modem Answer  Session   Dispatch
               00:10:00        never                      none     not set
                            Idle Session Disconnect Warning
                              never
Modem type is unknown.
Session limit is not set.
Time since activation: never
Editing is enabled.
History is enabled, history size is 10.
DNS resolution in show commands is enabled
Full user help is disabled
Allowed transports are pad telnet rlogin.  Preferred is telnet.
No output characters are padded
No special data dispatching characters
modem(slot/port)=1/0, csm_state(0x00000100)=CSM_IDLE_STATE, bchan_num=-1
modem_status(0x0000): VDEV_STATUS_UNLOCKED

Modem hardware state: CTS noDSR  DTR RTS
```

If you are having trouble:

- If you are having problems with making or receiving calls, make sure you turned on the protocols for connecting to the lines and configured for incoming and outgoing calls.
- If the calls are not coming up at all, turn on the **debug modem**, **debug modem csm**, and **debug isdn q931** commands to check for problems. When you finish viewing the messages, turn off the messages by entering the **no debug modem** command.

```
NAS# debug modem
NAS# debug modem csm
NAS# debug isdn q931
NAS# no debug modem
NAS# no debug modem csm
NAS# no debug isdn q931
```

- Enter the **debug modem ?** command for list of additional modem debugging commands:

```
NAS# debug modem ?
  b2b          Modem Special B2B
  csm          CSM activity
  maintenance  Modem maintenance activity
  mica         MICA Async driver debugging
  oob          Modem out of band activity
  tdm          B2B Modem/PRI TDM
  trace        Call Trace Upload
```

Create the Group Asynchronous Interface

Create a group asynchronous interface to project a set of core protocol characteristics to a range of asynchronous interfaces. Configuring the asynchronous interfaces as a group saves you time. Analog modem calls cannot enter the access server without this configuration.

To configure a group asynchronous interface, use the following commands beginning in global configuration mode:

Step	Command	Purpose
1	**interface group-async** *number*	Bring up a single group asynchronous interface.
2	**ip unnumbered loopback** *number*	To conserve IP addresses, configure the asynchronous interfaces as unnumbered, and assign the IP address of the loopback interface to them. [1]
3	**encapsulation ppp**	Enable PPP to run on the asynchronous interfaces in the group.
4	**async mode interactive**	Configure interactive mode on the asynchronous interface.
5	**ppp authentication chap pap** *list-name*	Enable CHAP and PAP authentication on the interface. Replace the *list-name* variable with a specified authentication list name.[2]
6	**peer default ip address pool** *poolname*	Assign dial-in clients IP addresses from an address pool. [3]
7	**no cdp enable**	Disable the Cisco Discovery Protocol on the interface.

Step	Command	Purpose
8	**group-range** *low-end-of-range high-end-of-range*	Specify the range of asynchronous interfaces to include in the group, which is usually equal to the number of modems you have in the access server.
9	**exit**	Return to global configuration mode.

1 You can also specify the Ethernet interface to conserver address space. In this case, enter the **ip unnumbered ethernet 0** command.
2 To create a string used to name the following list of authentication methods tried when a user logs in, see the **aaa authentication ppp** command. Authentication methods include RADIUS, TACACS+, Kerberos, etc.
3 To create an IP address pool, see the **ip local pool** global configuration command.

Verify the Group Interface Configuration

To verify the group interface configuration and check if one of the asynchronous interfaces are up, use the **show interface async** command:

```
NAS# show interface async 1
Async1 is up, line protocol is up
modem(slot/port)=1/0, csm_state(0x00000204)=CSM_IC4_CONNECTED, bchan_num=18
modem_status(0x0002): VDEV_STATUS_ACTIVE_CALL.

  Hardware is Async Serial
  Interface is unnumbered.  Using address of FastEthernet0 (10.1.1.10)
  MTU 1500 bytes, BW 115 Kbit, DLY 100000 usec, rely 255/255, load 1/255
  Encapsulation PPP, loopback not set, keepalive not set
  DTR is pulsed for 5 seconds on reset
  LCP Open
  Open: IPCP
  Last input 00:00:00, output 00:00:00, output hang never
  Last clearing of "show interface" counters never
  Queueing strategy: fifo
  Output queue 0/5, 0 drops; input queue 1/5, 0 drops
  5 minute input rate 37000 bits/sec, 87 packets/sec
  5 minute output rate 0 bits/sec, 0 packets/sec
     31063 packets input, 1459806 bytes, 0 no buffer
     Received 0 broadcasts, 0 runts, 0 giants, 0 throttles
     0 input errors, 0 CRC, 0 frame, 0 overrun, 0 ignored, 0 abort
     33 packets output, 1998 bytes, 0 underruns
     0 output errors, 0 collisions, 0 interface resets
     0 output buffer failures, 0 output buffers swapped out
     0 carrier transitions
```

If you are having trouble, enter the following debug commands and then send a call into the access server. Interpret the output and make configuration changes accordingly.

- **undebug all**
- **debug ppp negot**
- **debug ppp authentication**
- **debug modem**
- **debug ip peer**

```
nas-1# undebug all
All possible debugging has been turned off
nas-1# debug ppp negot
PPP protocol negotiation debugging is on
nas-1# debug ppp auth
PPP authentication debugging is on
nas-1# debug modem
Modem control/process activation debugging is on

nas-1# debug ip peer
IP peer address activity debugging is on
nas-1# show debug
General OS:
  Modem control/process activation debugging is on
Generic IP:
  IP peer address activity debugging is on
PPP:
  PPP authentication debugging is on
  PPP protocol negotiation debugging is on
nas-1#
*Mar  1 21:34:56.958: TTY4: DSR came up
*Mar  1 21:34:56.962: tty4: Modem: IDLE->READY
*Mar  1 21:34:56.970: TTY4: EXEC creation
*Mar  1 21:34:56.978: TTY4: set timer type 10, 30 seconds
*Mar  1 21:34:59.722: TTY4: Autoselect(2) sample 7E
*Mar  1 21:34:59.726: TTY4: Autoselect(2) sample 7EFF
*Mar  1 21:34:59.730: TTY4: Autoselect(2) sample 7EFF7D
*Mar  1 21:34:59.730: TTY4: Autoselect(2) sample 7EFF7D23
*Mar  1 21:34:59.734: TTY4 Autoselect cmd: ppp negotiate
*Mar  1 21:34:59.746: TTY4: EXEC creation
*Mar  1 21:34:59.746: TTY4: create timer type 1, 600 seconds
*Mar  1 21:34:59.786: ip_get_pool: As4: using pool default
*Mar  1 21:34:59.790: ip_get_pool: As4: returning address = 172.20.1.101
*Mar  1 21:34:59.794: TTY4: destroy timer type 1 (OK)
*Mar  1 21:34:59.794: TTY4: destroy timer type 0
*Mar  1 21:35:01.798: %LINK-3-UPDOWN: Interface Async4, changed state to up
*Mar  1 21:35:01.834: As4 PPP: Treating connection as a dedicated line
*Mar  1 21:35:01.838: As4 PPP: Phase is ESTABLISHING, Active Open
*Mar  1 21:35:01.842: As4 LCP: O CONFREQ [Closed] id 1 len 25
*Mar  1 21:35:01.846: As4 LCP:    ACCM 0x000A0000 (0x0206000A0000)
*Mar  1 21:35:01.850: As4 LCP:    AuthProto CHAP (0x0305C22305)
*Mar  1 21:35:01.854: As4 LCP:    MagicNumber 0x64E923A8 (0x050664E923A8)
*Mar  1 21:35:01.854: As4 LCP:    PFC (0x0702)
*Mar  1 21:35:01.858: As4 LCP:    ACFC (0x0802)
*Mar  1 21:35:02.718: As4 LCP: I CONFREQ [REQsent] id 3 len 23
*Mar  1 21:35:02.722: As4 LCP:    ACCM 0x000A0000 (0x0206000A0000)
*Mar  1 21:35:02.726: As4 LCP:    MagicNumber 0x00472467 (0x050600472467)
*Mar  1 21:35:02.726: As4 LCP:    PFC (0x0702)
*Mar  1 21:35:02.730: As4 LCP:    ACFC (0x0802)
*Mar  1 21:35:02.730: As4 LCP:    Callback 6  (0x0D0306)
*Mar  1 21:35:02.738: As4 LCP: O CONFREJ [REQsent] id 3 len 7
```

```
*Mar  1 21:35:02.738: As4 LCP:    Callback 6  (0x0D0306)
*Mar  1 21:35:02.850: As4 LCP: I CONFREQ [REQsent] id 4 len 20
*Mar  1 21:35:02.854: As4 LCP:    ACCM 0x000A0000 (0x0206000A0000)
*Mar  1 21:35:02.854: As4 LCP:    MagicNumber 0x00472467 (0x050600472467)
*Mar  1 21:35:02.858: As4 LCP:    PFC (0x0702)
*Mar  1 21:35:02.858: As4 LCP:    ACFC (0x0802)
*Mar  1 21:35:02.862: As4 LCP: O CONFACK [REQsent] id 4 len 20
*Mar  1 21:35:02.866: As4 LCP:    ACCM 0x000A0000 (0x0206000A0000)
*Mar  1 21:35:02.870: As4 LCP:    MagicNumber 0x00472467 (0x050600472467)
*Mar  1 21:35:02.870: As4 LCP:    PFC (0x0702)
*Mar  1 21:35:02.874: As4 LCP:    ACFC (0x0802)
*Mar  1 21:35:03.842: As4 LCP: TIMEout: State ACKsent
*Mar  1 21:35:03.842: As4 LCP: O CONFREQ [ACKsent] id 2 len 25
*Mar  1 21:35:03.846: As4 LCP:    ACCM 0x000A0000 (0x0206000A0000)
*Mar  1 21:35:03.850: As4 LCP:    AuthProto CHAP (0x0305C22305)
*Mar  1 21:35:03.854: As4 LCP:    MagicNumber 0x64E923A8 (0x050664E923A8)
*Mar  1 21:35:03.854: As4 LCP:    PFC (0x0702)
*Mar  1 21:35:03.858: As4 LCP:    ACFC (0x0802)
*Mar  1 21:35:03.962: As4 LCP: I CONFACK [ACKsent] id 2 len 25
*Mar  1 21:35:03.966: As4 LCP:    ACCM 0x000A0000 (0x0206000A0000)
*Mar  1 21:35:03.966: As4 LCP:    AuthProto CHAP (0x0305C22305)
*Mar  1 21:35:03.970: As4 LCP:    MagicNumber 0x64E923A8 (0x050664E923A8)
*Mar  1 21:35:03.974: As4 LCP:    PFC (0x0702)
*Mar  1 21:35:03.974: As4 LCP:    ACFC (0x0802)
*Mar  1 21:35:03.978: As4 LCP: State is Open
*Mar  1 21:35:03.978: As4 PPP: Phase is AUTHENTICATING, by this end
*Mar  1 21:35:03.982: As4 CHAP: O CHALLENGE id 1 len 26 from "nas-1"
*Mar  1 21:35:04.162: As4 CHAP: I RESPONSE id 1 len 26 from "krist"
*Mar  1 21:35:04.170: As4 AUTH: Started process 0 pid 47
*Mar  1 21:35:04.182: As4 CHAP: O SUCCESS id 1 len 4
*Mar  1 21:35:04.186: As4 PPP: Phase is UP
*Mar  1 21:35:04.190: As4 IPCP: O CONFREQ [Not negotiated] id 1 len 10
*Mar  1 21:35:04.194: As4 IPCP:    Address 172.20.1.2 (0x0306AC140102)
*Mar  1 21:35:04.202: As4 CDPCP: O CONFREQ [Closed] id 1 len 4
*Mar  1 21:35:04.282: As4 IPCP: I CONFREQ [REQsent] id 1 len 40
*Mar  1 21:35:04.282: As4 IPCP:    CompressType VJ 15 slots CompressSlotID (0x02
06002D0F01)
*Mar  1 21:35:04.286: As4 IPCP:    Address 0.0.0.0 (0x030600000000)
*Mar  1 21:35:04.290: As4 IPCP:    PrimaryDNS 0.0.0.0 (0x810600000000)
*Mar  1 21:35:04.294: As4 IPCP:    PrimaryWINS 0.0.0.0 (0x820600000000)
*Mar  1 21:35:04.298: As4 IPCP:    SecondaryDNS 0.0.0.0 (0x830600000000)
*Mar  1 21:35:04.302: As4 IPCP:    SecondaryWINS 0.0.0.0 (0x840600000000)
*Mar  1 21:35:04.306: As4 IPCP: O CONFREJ [REQsent] id 1 len 10
*Mar  1 21:35:04.310: As4 IPCP:    CompressType VJ 15 slots CompressSlotID (0x02
06002D0F01)
*Mar  1 21:35:04.314: As4 CCP: I CONFREQ [Not negotiated] id 1 len 15
*Mar  1 21:35:04.318: As4 CCP:    MS-PPC supported bits 0x00000001 (0x1206000000
01)
*Mar  1 21:35:04.318: As4 CCP:    Stacker history 1 check mode EXTENDED (0x11050
00104)
*Mar  1 21:35:04.322: As4 LCP: O PROTREJ [Open] id 3 len 21 protocol CCP
*Mar  1 21:35:04.326: As4 LCP:  (0x80FD0101000F12060000000111050001)
*Mar  1 21:35:04.330: As4 LCP:  (0x04)
*Mar  1 21:35:04.334: As4 IPCP: I CONFACK [REQsent] id 1 len 10
*Mar  1 21:35:04.338: As4 IPCP:    Address 172.20.1.2 (0x0306AC140102)
*Mar  1 21:35:04.342: As4 LCP: I PROTREJ [Open] id 5 len 10 protocol CDPCP (0x82
0701010004)
*Mar  1 21:35:04.342: As4 CDPCP: State is Closed
*Mar  1 21:35:05.186: %LINEPROTO-5-UPDOWN: Line protocol on Interface Async4, ch
anged state to up
*Mar  1 21:35:05.190: As4 PPP: Unsupported or un-negotiated protocol. Link cdp
*Mar  1 21:35:05.190: As4 PPP: Trying to negotiate NCP for Link cdp
*Mar  1 21:35:05.194: As4 CDPCP: State is Closed
*Mar  1 21:35:05.198: As4 CDPCP: TIMEout: State Closed
*Mar  1 21:35:05.202: As4 CDPCP: State is Listen
```

```
*Mar  1 21:35:06.202: As4 IPCP: TIMEout: State ACKrcvd
*Mar  1 21:35:06.206: As4 IPCP: O CONFREQ [ACKrcvd] id 2 len 10
*Mar  1 21:35:06.206: As4 IPCP:    Address 172.20.1.2 (0x0306AC140102)
*Mar  1 21:35:06.314: As4 IPCP: I CONFACK [REQsent] id 2 len 10
*Mar  1 21:35:06.318: As4 IPCP:    Address 172.20.1.2 (0x0306AC140102)
*Mar  1 21:35:07.274: As4 IPCP: I CONFREQ [ACKrcvd] id 2 len 34
*Mar  1 21:35:07.278: As4 IPCP:    Address 0.0.0.0 (0x030600000000)
*Mar  1 21:35:07.282: As4 IPCP:    PrimaryDNS 0.0.0.0 (0x810600000000)
*Mar  1 21:35:07.286: As4 IPCP:    PrimaryWINS 0.0.0.0 (0x820600000000)
*Mar  1 21:35:07.286: As4 IPCP:    SecondaryDNS 0.0.0.0 (0x830600000000)
*Mar  1 21:35:07.290: As4 IPCP:    SecondaryWINS 0.0.0.0 (0x840600000000)
*Mar  1 21:35:07.294: As4 IPCP: O CONFNAK [ACKrcvd] id 2 len 34
*Mar  1 21:35:07.298: As4 IPCP:    Address 172.20.1.101 (0x0306AC140165)
*Mar  1 21:35:07.302: As4 IPCP:    PrimaryDNS 172.20.5.100 (0x8106AC140564)
*Mar  1 21:35:07.306: As4 IPCP:    PrimaryWINS 172.20.5.101 (0x8206AC140565)
*Mar  1 21:35:07.310: As4 IPCP:    SecondaryDNS 172.20.6.100 (0x8306AC140664)
*Mar  1 21:35:07.314: As4 IPCP:    SecondaryWINS 172.20.6.101 (0x8406AC140665)
*Mar  1 21:35:07.426: As4 IPCP: I CONFREQ [ACKrcvd] id 3 len 34
*Mar  1 21:35:07.430: As4 IPCP:    Address 172.20.1.101 (0x0306AC140165)
*Mar  1 21:35:07.434: As4 IPCP:    PrimaryDNS 172.20.5.100 (0x8106AC140564)
*Mar  1 21:35:07.438: As4 IPCP:    PrimaryWINS 172.20.5.101 (0x8206AC140565)
*Mar  1 21:35:07.442: As4 IPCP:    SecondaryDNS 172.20.6.100 (0x8306AC140664)
*Mar  1 21:35:07.446: As4 IPCP:    SecondaryWINS 172.20.6.101 (0x8406AC140665)
*Mar  1 21:35:07.446: ip_get_pool: As4: validate address = 172.20.1.101
*Mar  1 21:35:07.450: ip_get_pool: As4: using pool default
*Mar  1 21:35:07.450: ip_get_pool: As4: returning address = 172.20.1.101
*Mar  1 21:35:07.454: set_ip_peer_addr: As4: address = 172.20.1.101 (3) is redun
dant
*Mar  1 21:35:07.458: As4 IPCP: O CONFACK [ACKrcvd] id 3 len 34
*Mar  1 21:35:07.462: As4 IPCP:    Address 172.20.1.101 (0x0306AC140165)
*Mar  1 21:35:07.466: As4 IPCP:    PrimaryDNS 172.20.5.100 (0x8106AC140564)
*Mar  1 21:35:07.470: As4 IPCP:    PrimaryWINS 172.20.5.101 (0x8206AC140565)
*Mar  1 21:35:07.474: As4 IPCP:    SecondaryDNS 172.20.6.100 (0x8306AC140664)
*Mar  1 21:35:07.474: As4 IPCP:    SecondaryWINS 172.20.6.101 (0x8406AC140665)
*Mar  1 21:35:07.478: As4 IPCP: State is Open
*Mar  1 21:35:07.490: As4 IPCP: Install route to 172.20.1.101
*Mar  1 21:35:25.038: As4 PPP: Unsupported or un-negotiated protocol. Link cdp
*Mar  1 21:36:12.614: TTY0: timer type 1 expired
*Mar  1 21:36:12.614: TTY0:  Exec timer (continued)
*Mar  1 21:36:25.038: As4 PPP: Unsupported or un-negotiated protocol. Link cdp
*Mar  1 21:37:25.038: As4 PPP: Unsupported or un-negotiated protocol. Link cdp
*Mar  1 21:38:25.038: As4 PPP: Unsupported or un-negotiated protocol. Link cdp
*Mar  1 21:39:25.038: As4 PPP: Unsupported or un-negotiated protocol. Link cdp
*Mar  1 21:40:25.038: As4 PPP: Unsupported or un-negotiated protocol. Link cdp
*Mar  1 21:41:25.038: As4 PPP: Unsupported or un-negotiated protocol. Link cdp
*Mar  1 21:42:25.038: As4 PPP: Unsupported or un-negotiated protocol. Link cdp
*Mar  1 21:43:25.038: As4 PPP: Unsupported or un-negotiated protocol. Link cdp
```

Monitor and Maintain Connections

This section describes the following monitoring and maintenance tasks that you can perform on asynchronous interfaces:

- Monitor and maintain asynchronous activity
- Debug asynchronous interfaces
- Debug PPP

To monitor and maintain asynchronous activity, use the following commands in privileged EXEC mode:

Command	Purpose
clear line *line-number*	Return a line to its idle state.
show async bootp	Display parameters that have been set for extended BOOTP requests.
show async status	Display statistics for asynchronous interface activity.
show line [*line-number*]	Display the status of asynchronous line connections.

To debug asynchronous interfaces, use the following debug command in privileged EXEC mode:

Command	Purpose
debug async {**framing** | **state** | **packets**}	Displays errors, changes in interface state, and log input and output.

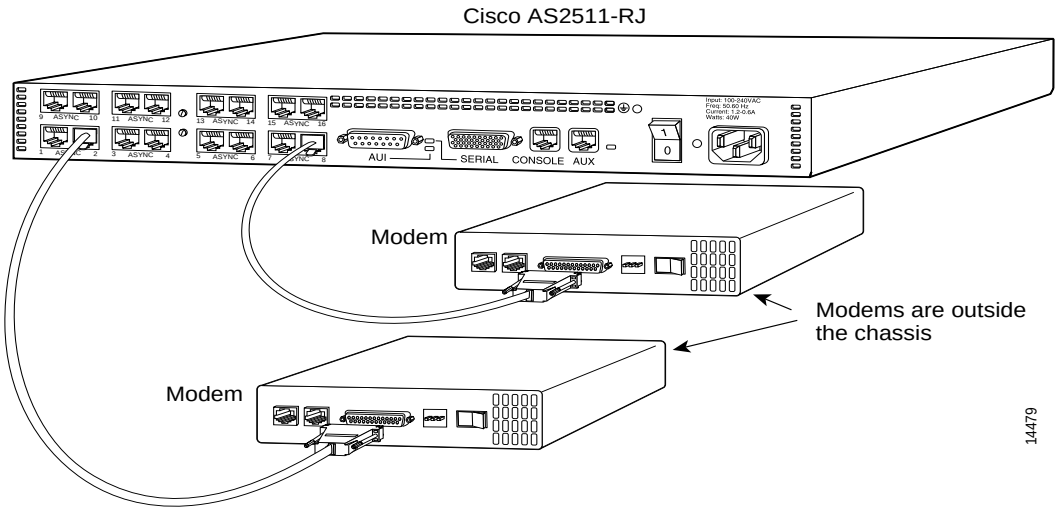
To debug PPP links, use the following debug commands in privileged EXEC mode:

Command	Purpose
debug ppp negotiation	Enable debugging of PPP protocol negotiation process.
debug ppp error	Display PPP protocol errors.
debug ppp packet	Display PPP packets sent and received.
debug ppp chap	Display errors encountered during remote or local system authentication.

Troubleshoot Externally Attached Modems

Some of Cisco's lower-end access servers have cable connections to external modems, such as the Cisco AS2511-RJ shown in Figure 18. Notice that the modems and modem cables are outside the chassis. The asynchronous interfaces and lines are inside the access server.

Figure 18 Cisco AS2511-RJ Access Server



This section describes how to do the following tasks:

- Verify the Line Speed and Modem Transmission Rate
- Communicate with the Modem
- Automatically Configure an External Modem
- Manually Configure an External Modem
- Test the Dial-In Connection

When you configure modems to function with your access server, you must provide initialization strings and other settings on the modem to tell it how to function with the access server.

For modem configuration information about specific access servers that have integrated modems, refer to the section “Check Other Modem Settings” later in this chapter.

Verify the Line Speed and Modem Transmission Rate

Make sure that the line speed on the access server matches the matches transmission rate, as shown in Table 5.

Table 5 Matching the Access Server Line Speed to Your Modem Speed

Modem Transmission Rate	Line Speed on the Access Server
9600	38400
14400	57600
28800	115200

Use the **show run** EXEC command. The line configuration fragment appears at the tail end of the output. In this example, lines 7 through 9 are transmitting at 115200 bps. There are 16 28.8 Kbps modems connected to a Cisco AS2511-RJ access server via a modem cable. See Figure 18.

```
as2511# show run
Building configuration...

Current configuration:
.
.
.
!
 line 1 16
  login local
  modem InOut
  speed 115200
  transport input all
  flowcontrol hardware
  script callback callback
  autoselect ppp
  autoselect during-login
 !
```

Communicate with the Modem

This section assumes you have already physically attached the modem to the access server. If not, refer to the user guide or installation and configuration guide for your access server for information about attaching modems.

Before you can configure the modem, you must establish communication with it, which requires terminal access to the modem's command environment. The process of manually configuring a modem consists of the following tasks:

- Establish a Direct Telnet Session to the Modem
- Test the Modem Connection
- Suspend and Terminate Telnet Sessions

Establish a Direct Telnet Session to the Modem

You communicate with the modem by establishing a direct Telnet session from the access server's asynchronous line, which is connected to the modem.

Note This process is also referred to as *reverse Telnet*. Performing a reverse Telnet means that you are initiating a Telnet session out the asynchronous line, instead of accepting a connection into the line (called a *forward* connection).

To establish a direct Telnet session to an external modem, determine the IP address of your LAN (Ethernet) interface, then enter a Telnet command to port 2000 + *n* on the access server, where *n* is the line number to which the modem is connected. For example, to connect to the modem attached to line 1, enter the following command from an EXEC session on the access server:

```
router# telnet 172.16.1.10 2001
Trying 172.16.1.10, 2001 ... Open
```

This example enables you to communicate with the modem on line 1 using the AT (attention) command set defined by the modem vendor.

Timesaver Use the **ip host** configuration command to simplify direct Telnet sessions with modems. The **ip host** command maps an IP address of a port to a device name. For example, the **modem1 2001 172.16.1.10** command enables you to enter **modem1** to initiate a connection with the modem, instead of repeatedly entering **telnet 172.16.1.10 2001** each time you want to communicate with the modem.

If you are unable to connect to the modem, check the following:

1. Issue the **show users** EXEC command. It should not indicate the line is in use.
2. Verify that the line is configured for **modem inout**.
3. Issue the **show line** EXEC command. The output should contain the following two lines:

   ```
   Modem state: Idle
   Modem hardware state: CTS noDSR  DTR RTS
   ```

4. Check to see if the virtual terminal connections to lines in the access server require passwords.
5. Check to see if the speed between the modem and the access server are the same. They are likely to be different. If they are different, switch off the modem, then switch it back on. This should match the speed of the modem with the speed of the access server.

Test the Modem Connection

After you make a direct Telnet connection to the modem, you need to test the connection. Send the modem the AT command to request its attention. It should respond with OK. For example:

```
at
OK
```

If the modem does not reply to the **AT** command, check the following:

1. Look at the output of the **show line 1** command. If it displays "no CTS" for the modem hardware state, the modem is not connected, powered on, and waiting for data, or the modem might not be configured for hardware flow control.
2. Check your cabling and the modem configuration (echo or result codes might be off). Enter the appropriate **AT** modem command to view the modem configuration, or enter the command **at&f** to return to factory defaults. Refer to your modem documentation to learn the appropriate **AT** command to view your modem configuration.

Suspend and Terminate Telnet Sessions

If you are connected to an external modem, the direct Telnet session must be terminated before the line can accept incoming calls. If you do not terminate the session, it will be indicated in the output of the **show users** command when it returns a modem state of ready if the line is still in use. If the line is no longer in use, the output of the **show line** *value* command will return a state of idle.

Terminating the Telnet session requires first suspending it, then disconnecting it. To suspend a Telnet session, enter the escape sequence **Ctrl-Shift-6 x** (press **Control-Shift-6**, let go, then press **x**). Enter the **disconnect** EXEC command to terminate the Telnet session.

Note Ensure that you can reliably issue the escape sequence to suspend a Telnet session. Some terminal emulation packages have difficulty sending the **Ctrl-Shift-6 x** sequence. Refer to your terminal emulation documentation for more information about escape sequences.

To suspend and then disconnect a Telnet session, perform the following steps:

Step 1 Enter **Ctrl-Shift-6 x** to suspend the Telnet session:

```
- suspend keystroke -
router#
```

Step 2 Enter the **where** EXEC command to check for open sessions:

```
router# where
Conn Host               Address             Byte  Idle Conn Name
*  1 172.16.1.10        172.16.1.10            0     0  172.16.1.10
   2 172.16.1.11        172.16.1.11            0    12  modem2
```

Step 3 After suspending a session with one modem, you can connect to another modem (then suspend it):

```
router# telnet modem2
Trying modem2 (172.16.1.11, 2002) ... Open

- suspend keystroke -
router#
```

Step 4 To disconnect (completely close) a session, issue the **disconnect** EXEC command:

```
router# disconnect line 1
Closing connection to 172.16.1.10 [confirm] y
router# disconnect line 2
Closing connection to 172.16.1.11 [confirm] y
router#
```

Note Before attempting to allow inbound connections, make sure you close all open connections to the modems attached to the access server. If you have a modem port in use, the modem will not accept a call properly.

After you have established and tested the connection to the modem, you can proceed with the next section "Automatically Configure an External Modem."

Automatically Configure an External Modem

The Cisco IOS software can issue initialization strings automatically for most types of modems externally attached to the access server. A modem initialization string is a series of parameter settings that are sent to your modem to configure it to interact with the access server in a specified way. The Cisco IOS software defines seven initialization strings that have been found to properly initialize most modems so that the modems function properly with Cisco access servers. These initialization strings have the following names:

- Codex_3260
- Usr_courier
- Usr_sportster
- Hayes_optima
- Global_village
- Viva
- Telebit_t3000

Note Internal or integrated modems, such as used by the Cisco AS5300, are preconfigured by Cisco Systems and do not need to be initialized.

If you do not know which of these modem strings is appropriate for your modems, issue the **modem autoconfigure discovery** line configuration command, as shown in the following example:

```
router# configure terminal
router(config)# line 1 16
router(config-line)# modem autoconfigure discovery
router(config-line)# Ctrl-Z
router# copy system:running-config nvram:startup-config
```

The Cisco IOS software first tries the first of these strings to see if the modem initializes properly. If not, the Cisco IOS software cycles to the next string and repeats the process until the appropriate string is found. If none of the strings properly initializes the modem, you must manually configure the modem (refer to "Manually Configure an External Modem" later in this chapter).

If you know that your modem can be configured using an initialization string from one of these scripts, you can issue the **modem autoconfigure type** *modem-name* command, where *modem-name* is one of the strings in the preceding list. If you list a specific modem type, initialization proceeds more quickly.

To display the list of modems for which the router has modem string entries, issue the **show modemcap** command. You can change a modem value that was returned from the **show modemcap** EXEC command. For example, you might want to add the factory default, **&F**, entry to the configuration file. To do this, enter the **modemcap edit** *modem-name attribute value* line configuration command. Configure one attribute of one modem at a time.

The following example shows how to enter line configuration mode and issue the **modem autoconfigure type** *modem-name* command for a US Robotics Sportster modem:

```
router(config-line)# modem autoconfigure type usr_sportster
```

For more information about the recommended strings for any type of modem, refer to the section "Sample Modem Strings" in the appendix "Configuring Modem Support and Chat Scripts" in the Dial Solutions Command Reference.

Manually Configure an External Modem

If you cannot configure your modem automatically, you must configure it manually. The following sections describe how to configure your externally attached modem manually:

- Configure Modem Initialization Strings
- Check Other Modem Settings
- Initialize the Modem

Configure Modem Initialization Strings

This section describes how to determine and issue the correct initialization string for your modem and configure your modem with it.

Modem command sets vary widely. Although most modems use the Hayes command set (prefixing commands with **AT**), Hayes-compatible modems do not use identical **AT** command sets.

Refer to your modem manufacturer's documentation to learn how to examine the current and stored configuration of the modem you are using. Generally, you enter **AT** commands such as **&v**, **i4**, or ***o** to view, inspect, or observe the settings.

Note You must first create a direct Telnet or connection session to the modem before you can send an initialization string. You can use **AT&F** as a basic modem initialization string in most cases.

A sample modem initialization string for a US Robotics Courier modem is as follows:

```
&b1&h1&r2&c1&d3&m4&k1s0=1
```

Modem initialization strings enable the following functions:

- Locks the speed of the modem to the speed of the serial port on the access server
- Sets Hardware Flow Control (RTS/CTS)
- Ensures Correct DCD Operation
- Ensures Proper DTR Interpretation
- Answers Calls on the First Ring

Timesaver Initialization strings for other modems are listed in the appendix "Configuring Modem Strings and Chat Scripts" in the *Dial Solutions Command Reference*.

Note Make sure to turn off automatic baud rate detection because the modem speeds must be set to a fixed value.

The port speed must not change when a session is negotiated with a remote modem. If the speed of the port on the access server is changed, you must establish a direct Telnet session to the modem and send an **AT** command so that the modem can learn the new speed.

Modems differ in the method they use to lock the EIA/TIA-232 (serial) port speed. In the modem documentation, vendors use terms such as, port-rate adjust, speed conversion, or buffered mode. Enabling error correction often puts the modem in the buffered mode. Refer to your modem documentation to see how your modem locks speed (check the settings **&b**, **\j**, **&q**, **\n**, or s-register settings).

Ready-To-Send (RTS) and Clear-To-Send (CTS) signals must be used between the modem and the access server to control the flow of data. Incorrectly configuring flow control for software or setting no flow control can result in hung sessions and loss of data. Modems differ in the method they use to enable hardware flow control. Refer to your modem documentation to see how to enable hardware flow control (check the settings **&e**, **&k**, **&h**, **&r**, or s-register).

The modem must use the data carrier detect (DCD) wire to indicate to the access server when a session has been negotiated and is established with a remote modem. Most modems use the setting **&c1**. Refer to your modem documentation for the DCD settings used with your modem.

The modem must interpret a toggle of the Data Terminal Ready (DTR) signal as a command to drop any active call and return to the stored settings. Most modems use the settings **&d2** or **&d3**. Refer to your modem documentation for the DTR settings used with your modem.

If a modem is used to service incoming calls, it must be configured to answer a call after a specific number of rings. Most modems use the setting **s0=1** to answer the call after one ring. Refer to your modem documentation for the settings used with your modem.

Check Other Modem Settings

This section defines other settings that might be needed or desirable depending on your modem.

Error correction can be negotiated between two modems to ensure a reliable data link. Error correction standards include LAPM and MNP4. V.42 error correction allows either LAPM or MNP4 error correction to be negotiated. Modems differ in the way they enable error correction. Refer to your modem documentation for the error correction methods used with your modem.

Data compression can be negotiated between two modems to allow for greater data throughput. Data compression standards include V.42 bis and MNP5. Modems differ in the way they enable data compression. Refer to your modem documentation for the data compression settings used with your modem.

Initialize the Modem

Refer to this section if you could not or chose not to initialize your modems automatically, as described in the "Automatically Configure an External Modem" section earlier in this chapter.

After the modem initialization string has been determined, perform the following steps to configure the modem. The steps below configure a U.S. Robotics Courier modem on line 1 (decimal number 2000 + line number 1 = 2001).

Step 1 Map a host name to a decimal port. The port number is 200x, plus the number of the TTY line. The following example maps port 2001 to the IP address of the Ethernet0 interface on the access server (172.16.1.10):

```
router(config)# ip host modem1 2001 172.16.1.10
router(config)# exit
router#
```

Step 2 Establish a direct Telnet session to the modem:

```
router# telnet modem1
Trying modem1 (172.16.1.10, 2001)... Open
```

Step 3 Return the modem to its factory defaults (this step is optional):

```
at&f
OK
```

Step 4 Configure the modem with an initialization string. The following example string is for a U.S. Robotics Courier modem:

```
at&b1&h1&r2&c1&d3&m4&k1s0=1
OK
```

Step 5 Store the modem settings in NVRAM on the modem:

```
at&w
OK
```

Note Some modems need to be "strapped" so that they start up with saved settings when powered on, rather than using defaults. You should make sure your modem is strapped accordingly.

Step 6 Suspend and disconnect your Telnet session:

```
- suspend keystroke -
router# disconnect
Closing connection to modem1 [confirm] y
router#
```

Timesaver The **script-reset** line configuration command can automate the configuration of your modems. See the "Technical Tips" section on CCO for more information.

Test the Dial-In Connection

The access server and modem are now correctly configured for dial-in access. Before configuring any additional protocols for the line (such as SLIP, PPP, or ARA), test the dial-in connection.

Note The same configuration issues exist between the client data terminal equipment (DTE) and client modem. Make sure you have the correct EIA/TIA-232 cabling and modem initialization string for your client modem.

The following is an example of a successful connection from a PC using a U.S. Robotics Courier modem to dial in to a Cisco 2500 series access server:

```
at&f&c1&d3&h1&r2&b1&m4&k1&w
OK
atdt9,5551234
CONNECT 14400/ARQ/V32/LAPM/V42BIS
User Access Verification
Username: janedoe
Password:
router>
```

Set Up Advanced Features

The following modem line characteristics and modem features are discussed in the following sections:

- Background Information
- Configure Automatic Dialing
- Automatically Answer a Modem
- Support Dial-In and Dial-Out Connections
- Configure a Line Timeout Interval
- Close Modem Connections
- Configure a Line to Automatically Disconnect
- Support Old-Style Dial-In Modems

- Support Reverse Modem Connections and Prevent Incoming Calls
- Configure Support for Async BOOTP Requests
- Configure Dual-Purpose Ports
- Conserve Network Addresses
- Use Advanced Addressing Methods for Remote Devices
- Configure Dedicated or Interactive PPP and SLIP Sessions
- Enable Routing on Asynchronous Interfaces
- Establish and Control the EXEC Process
- Configure the Auxiliary (AUX) Port
- Configure Autoselect and Verify that It Works
- Define a Command String for Automatic Execution
- Configure Rotary Groups
- Specify Decimal TCP Port Numbers when Connecting to Lines
- Optimize Available Bandwidth

Background Information

This section provides information to help you understand modem signaling, flow control, and line states.

About Signals and Flow Control

The EIA/TIA-232 output signals are Transmit Data (TXDATA), Data Terminal Ready (DTR), and Ready To Send (RTS, 2500 only). The input signals are Receive Data (RXDATA), Clear to Send (CTS), and RING. The sixth signal is ground. Depending on the type of modem control your modem uses, these names may or may not correspond to the standard EIA/TIA-232 signals.

Dial-up modems that operate over normal telephone lines at speeds of 28800 bps use hardware flow control to stop the data from reaching the host by toggling an EIA/TIA-232 signal when their limit is reached.

In addition to hardware flow control, modems require special software configuring. For example, they must be configured to create an EXEC session when a user dials in and to hang up when the user exits the EXEC. These modems also must be configured to close any existing network connections if the telephone line hangs up in the middle of a session.

The Cisco IOS software supports hardware flow control on its CTS input signal, which is also used by the normal modem handshake.

Signal and Line State Diagrams

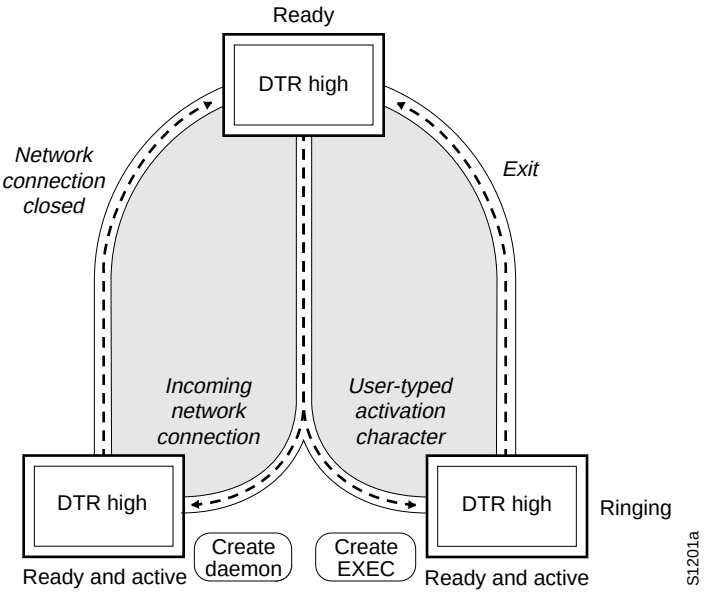
Signal and line state diagrams accompany some of the tasks in the following sections to illustrate how the modem control works. The diagrams show two processes:

- The “create daemon” process creates a TTY daemon that handles the incoming network connection.
- The “create EXEC” process creates the process that interprets user commands. (Refer to Figure 19 through Figure 24.)

In the diagrams, the current signal state and the signal the line is watching are listed inside each box. The state of the line (as displayed by the **show line** EXEC command) is listed next to the box. Events that change that state appear in italics along the event path, and actions that the software performs are described within the ovals.

Figure 19 illustrates line states when no modem control is set. The DTR output is always high, and CTS and RING are completely ignored. The Cisco IOS software starts an EXEC session when the user types the activation character. Incoming TCP connections occur instantly if the line is not in use and can be closed only by the remote host.

Figure 19 EXEC and Daemon Creation on a Line with No Modem Control



Configure Automatic Dialing

With the dial-up capability, you can set a modem to dial the phone number of a remote router automatically. This feature offers cost savings because phone line connections are made only when they are needed—you only pay for using the phone line when there is data to be received or sent. To configure a line for automatic dialing, use the following command in line configuration mode:

Command	Purpose
modem dtr-active	Configure a line to initiate automatic dialing.

Using the **modem dtr-active** command causes a line to raise DTR signal only when there is an outgoing connection (such as reverse Telnet, NASI, or DDR), rather than leave DTR raised all the time. When raised, DTR potentially tells the modem that the router is ready to accept a call.

Automatically Answer a Modem

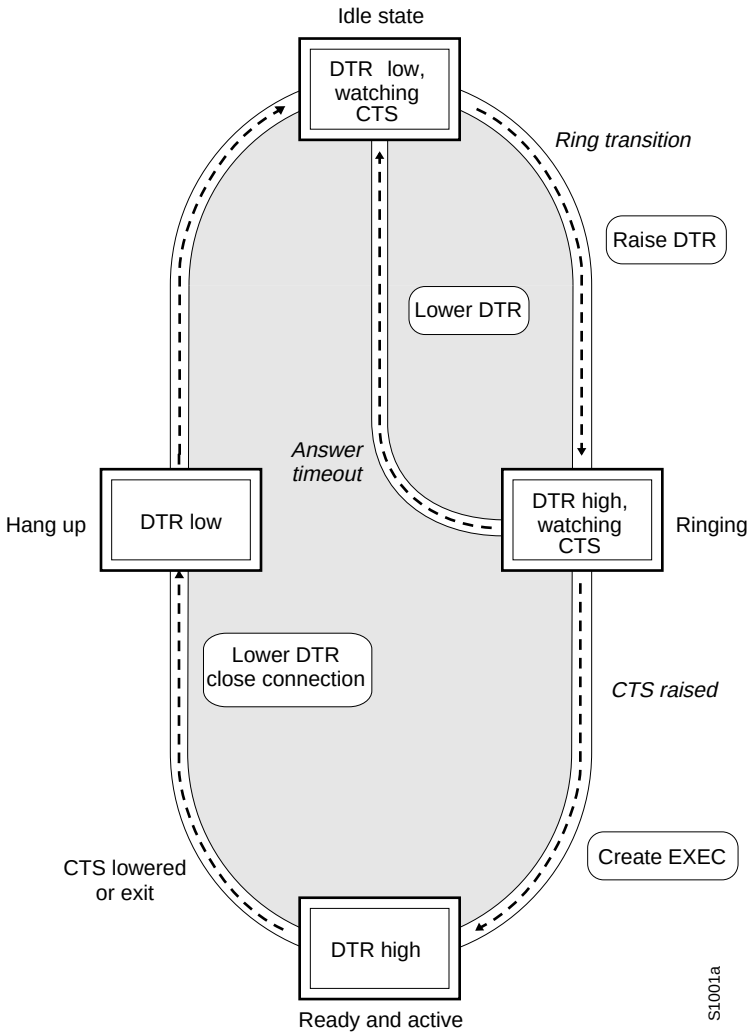
You can configure a line to answer a modem automatically. You also can configure the modem to answer the telephone on its own (as long as DTR is high), drop connections when DTR is low, and use its Carrier Detect (CD) signal to accurately reflect the presence of carrier. (Configuring the modem is a modem-dependent process.) Wire the modem's CD signal (generally pin-8) to the router's RING input (pin-22), and use the following command in line configuration mode:

Command	Purpose
modem dialin	Configure a line to automatically answer a modem.

You can turn on the modem's hardware flow control independently to respond to the status of the router's CTS input. Wire CTS to whatever signal the modem uses for hardware flow control. If the modem expects to control hardware flow in both directions, you might also need to wire the modem's flow control input to some other signal that the router always has high (such as the DTR signal).

Figure 20 illustrates the **modem dialin** process with a high-speed dial-up modem. When the Cisco IOS software detects a signal on the RING input of an idle line, it starts an EXEC or autobaud process on that line. If the RING signal disappears on an active line, the Cisco IOS software closes any open network connections and terminates the EXEC facility. If the user exits the EXEC or the software terminates because of no user input, the line makes the modem hang up by lowering the DTR signal for five seconds. After five seconds, the modem is ready to accept another call.

Figure 20 EXEC Creation on a Line Configured for a High-Speed Modem



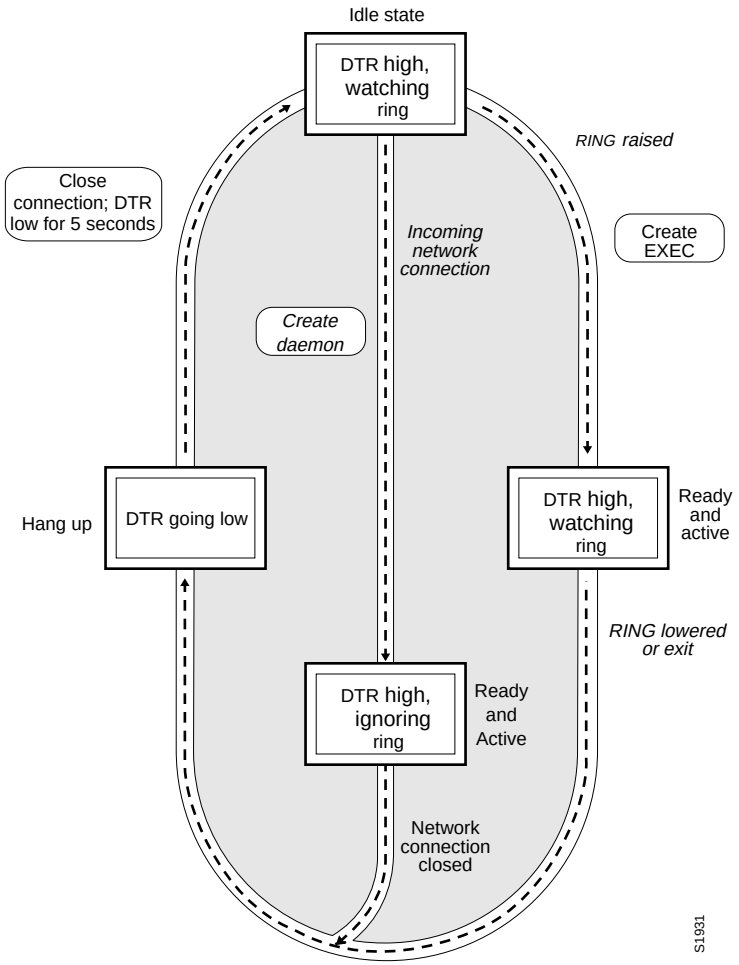
Support Dial-In and Dial-Out Connections

You can configure a line for both incoming and outgoing calls by using the following command in line configuration mode:

Command	Purpose
modem inout	Configure a line for both incoming and outgoing calls.

Figure 21 illustrates the **modem inout** command. If the line is activated by raising the data set ready (DSR) signal, it functions exactly as a line configured with the **modem dialin** line configuration command described in the "Automatically Answer a Modem" section earlier in this chapter. If the line is activated by an incoming TCP connection, the line functions similarly to lines not used with modems.

Figure 21 EXEC and Daemon Creation for Incoming and Outgoing Calls



Note If your system incorporates dial-out modems, consider using access lists to prevent unauthorized use.

Configure a Line Timeout Interval

You can change the interval that the Cisco IOS software waits for the CTS signal after raising the DTR signal in response to the DSR (the default is 15 seconds). To do so, use the following command in line configuration mode. The timeout applies to the **modem callin** command only.

Command	Purpose
modem answer-timeout *seconds*	Configure modem line timing.

Note The DSR signal is called RING on older ASM-style chassis.

Close Modem Connections

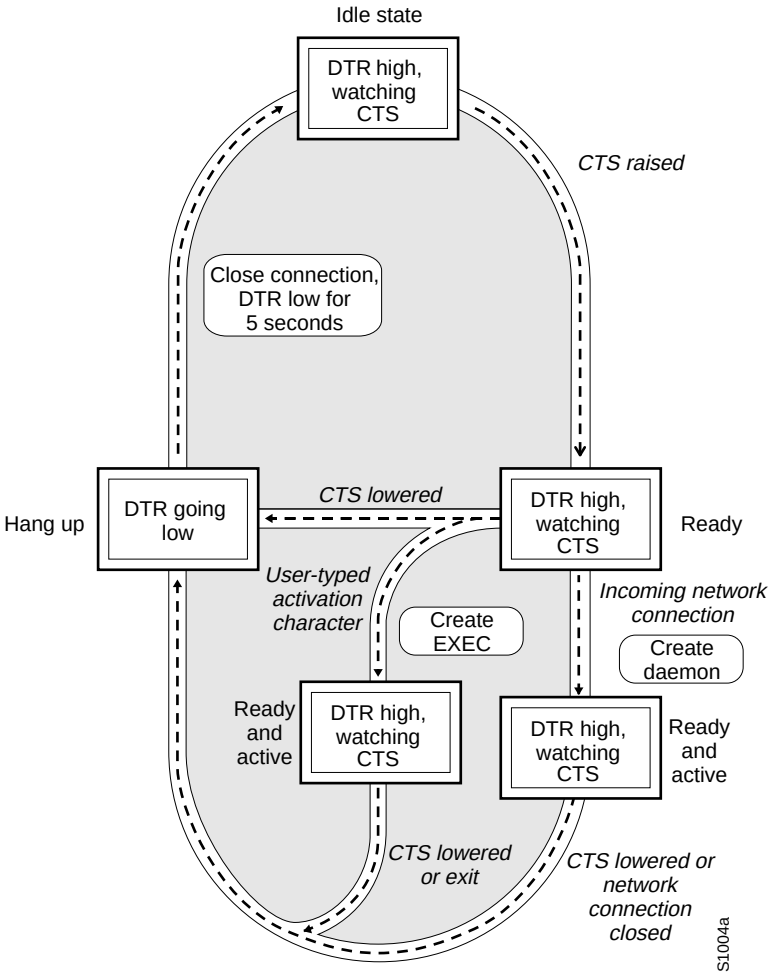
You can configure a line to close connections from a user's terminal when the terminal is turned off and prevent inbound connections to devices that are out of service. To do so, use the following command in line configuration mode:

Command	Purpose
modem cts-required	Configure a line to close connections.

Figure 22 illustrates the **modem cts-required** command operating in the context of a continuous CTS signal. This form of modem control requires that the CTS signal be high for the entire session. If CTS is not high, the user's input is ignored and incoming connections are refused (or sent to the next line in a rotary group).

Note For the Cisco IOS software to reliably detect a CTS signal change, the CTS signal must remain in the new state for at least one full second.

Figure 22 EXEC and Daemon Creation on a Line Configured for Continuous CTS



Configure a Line to Automatically Disconnect

You can configure automatic line disconnect by using the following command in line configuration mode:

Command	Purpose
autohangup	Configure automatic line disconnect.

The **autohangup** command causes the EXEC facility to issue the **exit** command when the last connection closes. This feature is useful for UNIX-to-UNIX copy program (UUCP) applications because UUCP scripts cannot issue a command to hang up the telephone. This feature is not often used.

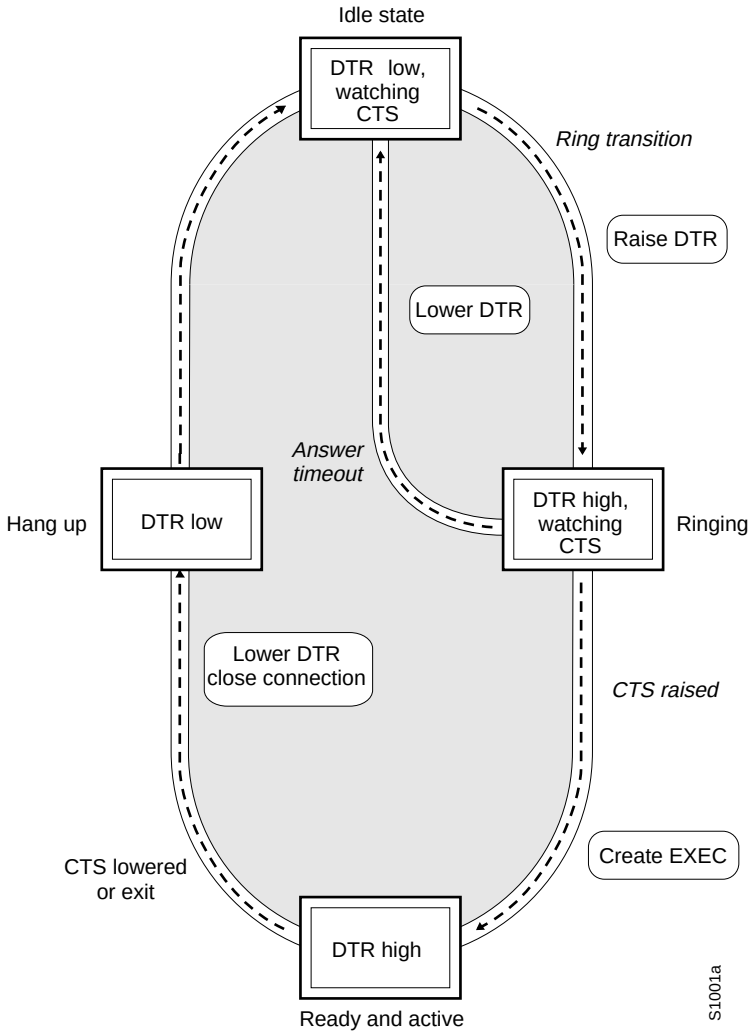
Support Old-Style Dial-In Modems

The Cisco IOS software supports dial-in modems that use DTR to control the off-hook status of the telephone line. This feature is supported primarily on old-style modems, especially those in Europe. To configure the line to support this feature, use the following command in line configuration mode:

Command	Purpose
modem callin	Configure a line for a dial-in modem.

Figure 23 illustrates the **modem callin** command. When a modem dialing line is idle, it has its DTR signal at a low state and waits for a transition to occur on the DSR (RING) input. This transition causes the line to raise the DTR signal and start watching the CTS signal from the modem. After the modem raises CTS, the Cisco IOS software creates an EXEC session on the line. If the timeout interval (set with the **modem answer-timeout** command) passes before the modem raises the CTS signal, the line lowers the DTR signal and returns to the idle state.

Figure 23 EXEC Creation on a Line Configured for Modem Call-In



Note The **modem callin** and **modem cts-required** line configuration commands are useful for SLIP operation. These commands ensure that when the line is hung up or the CTS signal drops, the line reverts from SLIP mode to normal interactive mode. These commands do not work if you put the line in network mode permanently.

Although you can use the **modem callin** line configuration command with newer modems, the **modem dialin** line configuration command described in this section is more appropriate. The **modem dialin** command frees up CTS input for hardware flow control. Modern modems do not require the assertion of DTR to answer a phone line (that is, to take the line off-hook).

Support Reverse Modem Connections and Prevent Incoming Calls

In addition to initiating connections, the Cisco IOS software can receive incoming connections. This capability allows you to attach serial and parallel printers, modems, and other shared peripherals to the router or access server and drive them remotely from other modem-connected systems. The Cisco IOS software supports reverse TCP, XRemote, and LAT connections.

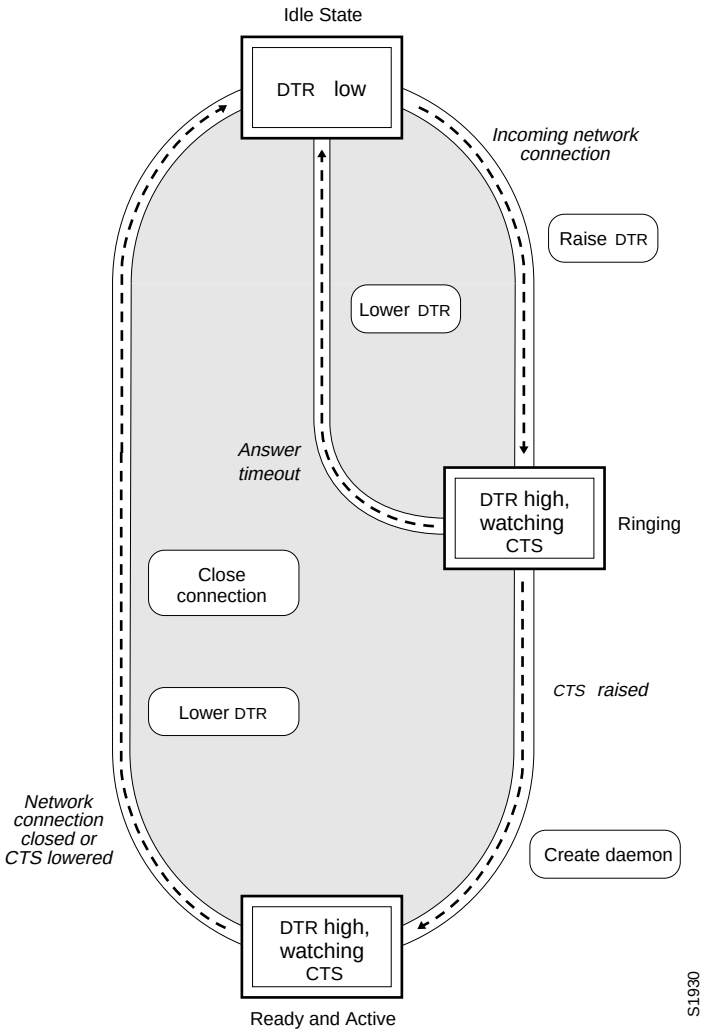
The specific TCP port or socket, to which you attach the device determines the type of service that the Cisco IOS software provides on a line. When you attach the serial lines of a computer system or a data terminal switch to the serial lines of the access server, the access server can act as a network front-end device for a host that does not support the TCP/IP protocols. This arrangement is sometimes called *front-ending*, or *reverse connection mode*.

The Cisco IOS software supports ports connected to computers that are connected to modems. You can configure the Cisco IOS software to function somewhat like a modem by using the following command in line configuration mode. This command also prevents incoming calls.

Command	Purpose
modem callout	Configure a line for reverse connections and prevent incoming calls.

Figure 24 illustrates the **modem callout** process. When the Cisco IOS software receives an incoming connection, it raises the DTR signal and waits to see if the CTS signal is raised to indicate that the host has noticed the router's DTR signal. If the host does not respond within the interval set by the **modem answer-timeout** line configuration command, the software lowers the DTR signal and drops the connection.

Figure 24 Daemon Creation on a Line Configured for Modem Call-out



Configure Support for Async BOOTP Requests

To configure Cisco IOS software to respond to BOOTP requests from client machines, use the following command in global configuration mode:

Command	Purpose
async-bootp *tag* [**:***hostname*] *data*	Specify the router network information that is sent in response to BOOTP requests.

Configure Dual-Purpose Ports

To specify the mode of a low-speed serial interface as either synchronous or asynchronous, use the following command in interface configuration mode:

Command	Purpose
physical-layer {**sync** | **async**}	Specify the mode of a low-speed interface as either synchronous or asynchronous.

This command applies only to low-speed serial interfaces available on Cisco 2520 through Cisco 2523 routers.

In synchronous mode, low-speed serial interfaces support all interface configuration commands available for high-speed serial interfaces, except the following two commands:

- **sdlc cts-delay**
- **sdlc rts-timeout**

When placed in asynchronous mode, low-speed serial interfaces support all commands available for standard asynchronous interfaces. The default is synchronous mode.

Note that when you enter this command, it does not appear in the output of **more system:running-config** and **more nvram:startup-config** command, because the command is a physical-layer command.

Conserve Network Addresses

When asynchronous routing is enabled, you might find it necessary to conserve network addresses by configuring the asynchronous interfaces as *unnumbered*. An unnumbered interface does not have an address. Network resources are therefore conserved because fewer network numbers are used and routing tables are smaller.

To configure an unnumbered interface, use the following command in interface configuration mode:

Command	Purpose
ip unnumbered *type number*	Conserve IP addresses by configuring the asynchronous interfaces as unnumbered, and assign the IP address of the interface type that you want to leverage.

Whenever the unnumbered interface generates a packet (for example, a routing update), it uses the address of the specified interface as the source address of the IP packet. It also uses the address of the specified interface to determine which routing processes are sending updates over the unnumbered interface.

You can use the IP unnumbered feature even if the system on the other end of the asynchronous link does not support it. The IP unnumbered feature is transparent to the other end of the link because each system bases its routing activities on information in the routing updates it receives and on its own interface address.

Use Advanced Addressing Methods for Remote Devices

You can control whether addressing is dynamic (the user specifies the address at the EXEC level when making the connection), or whether default addressing is used (the address is forced by the system). If you specify dynamic addressing, the router must be in interactive mode and the user will enter the address at the EXEC level.

It is common to configure an asynchronous interface to have a default address and to allow dynamic addressing. With this configuration, the choice between the default address or a dynamic addressing is made by the users when they enter the **slip** or **ppp** EXEC command. If the user enters an address, it is used, and if the user enters the **default** keyword, the default address is used.

This section describes the following tasks:

- Assign a default asynchronous address.
- Allow an asynchronous address to be assigned dynamically.

Assign a Default Asynchronous Address

Use the following command in interface configuration mode to assign a permanent default asynchronous address:

Command	Purpose
peer default ip address *ip-address*	Assign a default IP address to an asynchronous interface.

Use the **no** form of this command to disable the default address. If the server has been configured to authenticate asynchronous connections, you are prompted for a password after entering the **slip default** or **ppp default** EXEC command before the line is placed into asynchronous mode.

The assigned default address is implemented when the user enters the **slip default** or **ppp default** EXEC command. The transaction is validated by the Terminal Access Controller Access System (TACACS) server, when enabled, and the line is put into network mode using the address that is in the configuration file.

Configuring a default address is useful when the user is not required to know the IP address to gain access to a system (for example, users of a server that is available to many students on a campus). Instead of requiring each user to know an IP address, they only need to enter the **slip default** or **ppp default** EXEC command and let the server select the address to use. See the chapter "Making Connections to Network Devices" in this book for more information about the **slip** and **ppp** EXEC commands.

Allow an Asynchronous Address to Be Assigned Dynamically

When a line is configured for dynamic assignment of asynchronous addresses, the user enters the **slip** or **ppp** EXEC command and is prompted for an address or logical host name. The address is validated by TACACS, when enabled, and the line is assigned the given address and put into asynchronous mode. Assigning asynchronous addresses dynamically is also useful when you want to assign set addresses to users. For example, an application on a personal computer that automatically dials in using SLIP and polls for electronic mail messages can be set up to dial in periodically and enter the required IP address and password.

To assign asynchronous addresses dynamically, use the following command in interface configuration mode:

Command	Purpose
async dynamic address	Allow the IP address to be assigned when the protocol is initiated.

The dynamic addressing features of the internetwork allow packets to get to their destination and back regardless of the access server, router, or network they are sent from. For example, if a host such as a laptop computer moves from place to place it can keep the same address no matter where it is dialing in from.

Logical host names are first converted to uppercase and then sent to the TACACS server for authentication.

Configure Dedicated or Interactive PPP and SLIP Sessions

You can configure one or more asynchronous interfaces on your access server (and one on a router) to be in dedicated network interface mode. In dedicated mode, an interface is automatically configured for SLIP or PPP connections. There is no user prompt or EXEC level, and no end-user commands are required to initiate remote-node connections. If you want a line to be used only for SLIP or PPP connections, configure the line for dedicated mode.

In interactive mode, a line can be used to make any type of connection, depending on the EXEC command entered by the user. For example, depending on its configuration, the line could be used for Telnet or XRemote connections, or SLIP or PPP encapsulation. The user is prompted for an EXEC command before a connection is initiated.

You can configure an asynchronous interface to be in dedicated network mode. When the interface is configured for dedicated mode, the end user cannot change the encapsulation method, address, or other parameters.

To configure an interface for dedicated network mode or return it back to interactive mode, use one of the following commands in interface configuration mode.

Command	Purpose
async mode dedicated	Place the line into dedicated asynchronous network mode.
async mode interactive	Return the line to interactive mode.

By default, no asynchronous mode is configured. In this state, the line is not available for inbound networking because the SLIP and PPP connections are disabled.

Enable Routing on Asynchronous Interfaces

To route IP packets on an asynchronous interface, use one of the following commands in interface configuration mode:

Command	Purpose
async dynamic routing	Configure an asynchronous interface for dynamic routing. Use this command to manually bring up PPP from an EXEC session.
async default routing	Automatically configure an asynchronous interface for routing. Use this command to enable two routers to communicate over an async dial backup link.

This **async dynamic routing** command routes IP packets on an asynchronous interface, which permits you to enable the IGRP, RIP, and OSPF routing protocols for use when the user makes a connection using the **ppp** or **slip** EXEC commands. The user must, however, specify the **/routing** keyword at the SLIP or PPP command line.

For asynchronous interfaces in interactive mode, the **async default routing** command causes the **ppp** and **slip** EXEC commands to be interpreted as though the **/route** switch had been included in the command. For asynchronous interfaces in dedicated mode, the **async dynamic routing** command enables routing protocols to be used on the line. Without the **async default routing** command, there is no way to enable the use of routing protocols automatically on a dedicated asynchronous interface.

Establish and Control the EXEC Process

By default, the Cisco IOS software starts an EXEC process on all lines. However, you can control EXEC processes, as follows:

- Turn the EXEC process on or off.

  A serial printer, for example, should not have an EXEC session started.

- Set the idle terminal timeout interval.

The EXEC command interpreter waits for a specified amount of time to receive user input. If no input is detected, the EXEC facility resumes the current connection. If no connections exist, it returns the terminal to the idle state and disconnects the incoming connection. To control the EXEC process, use the following commands in line configuration mode:

Step	Command	Purpose
1	**exec**	Turn on EXEC processes.
2	**exec-timeout** *minutes* [*seconds*]	Set the idle terminal timeout interval.

Configure the Auxiliary (AUX) Port

The AUX port is typically configured as an asynchronous serial interface on routers without built-in asynchronous interfaces. To configure the AUX port as an asynchronous interface, configure it first as an auxiliary line with the **line aux 1** global configuration command.

The AUX port sends a DTR signal only when a Telnet connection is established. The auxiliary port does not use Ready to Send/Clear to Send (RTS/CTS) handshaking for flow control. To understand the differences between standard asynchronous interfaces and AUX ports configured as an asynchronous interface, refer to Table 6. To enable the auxiliary port, use the following command in global configuration mode:

Command	Purpose
line aux *line-number*	Enable the auxiliary serial DTE port.

You cannot use the auxiliary (AUX) port as a second console port. To use the AUX port as a console port, you must order a special cable from your technical support personnel.

On an access server, you can configure any of the available asynchronous interfaces (1 through 8, 16, or 48). The auxiliary port (labeled AUX on the back of the product) can also be configured as an asynchronous serial interface, although performance on the AUX port is much slower than on standard asynchronous interfaces and does not support some features. Table 6 illustrates why asynchronous interfaces permit substantially better performance than AUX ports configured as asynchronous interfaces.

Table 6 Differences between the Auxiliary (AUX) Port and the Asynchronous Port

Feature	Asynchronous Interface	Auxiliary Port
Maximum speed	115200 kbps	38400 kbps
Supports DMA buffering[1]	Yes	No
PPP framing on chip[2]	Yes	No
IP fast switching[3]	Yes	No

1 Direct Memory Access (DMA) buffering moves data packets directly to and from system memory without interrupting the main central processing unit (CPU). This process removes overhead from the CPU and increases overall system performance.

2 PPP framing on a hardware chip removes overhead from the router's CPU, which enables the router to sustain 115.2 kbps throughput on all asynchronous ports simultaneously.

3 After the destination of the first IP packet is added to the fast switching cache, it is fast switched to and from other interfaces with minimal involvement from the main processor.

On routers without built-in asynchronous interfaces, only the AUX port can be configured as an asynchronous serial interface. To configure the AUX port as an asynchronous interface, you must also configure it as an auxiliary line with the **line aux 1** command. Access servers do not have this restriction. Use the line command with the appropriate line configuration commands for modem control, such as speed.

Only IP packets can be sent across lines configured for SLIP. PPP supports transmission of IP, IPX, and AppleTalk packets on an asynchronous serial interface.

Configure Autoselect and Verify that It Works

Autoselect is used by the access server to sense the protocol being received on an incoming line and launch the appropriate protocol. Autoselect can be used for AppleTalk Remote Access (ARA), Point-to-Point Protocol (PPP), or Serial Line Internet Protocol (SLIP).

When using Autoselect, “login” authentication is by-passed, so if security is required it must be performed at the protocol level (i.e. ARAP or PPP authentication). SLIP does not offer protocol layer authentication.

To configure the Cisco IOS software to allow and ARA, PPP, SLIP session to start automatically, use the following command in line configuration mode:

Command	Purpose
autoselect {**arap** | **ppp** | **slip** | **during login**}	Configure a line to automatically start an ARA, PPP, or SLIP session.

The **autoselect** command enables the Cisco IOS software to start a process automatically when a start character is received.

The **autoselect** command bypasses the login prompt and enables the specified session to begin automatically. However, by entering the **autoselect** command with the **during login** keyword, the username or password prompt appears without needing to press the Return key, thus "login" users will get a decent prompt instead of needing to press the Return key. While the username or password prompt is displayed, you can choose either to answer these prompts or to send packets from an autoselected protocol.

Normally a router avoids line and modem noise by clearing the initial data received within the first one or two seconds. However, when the autoselect PPP feature is configured, the router flushes characters initially received and then waits for more traffic. This flush causes time out problems with applications that send only one carriage return. To ensure that the input data sent by a modem or other asynchronous device is not lost after line activation, enter the **flush-at-activation** line configuration command.

Note When you use the **autoselect** command, the activation character should be set to the default Return, and exec-character-bits to 7. If you change these defaults, the application cannot recognize the activation request.

Verify Autoselect PPP

The following trace appears when the **debug modem** command and **debug ppp negotiation** command are enabled. As PPP calls pass through the access server, you should see this output.

When using autoselect, "login" authentication is by-passed. If security is required it must be performed at the protocol level (i.e. ARAP or PPP authentication). SLIP does not offer protocol layer authentication.

```
22:21:02: TTY1: DSR came up
22:21:02: tty1: Modem: IDLE->READY
22:21:02: TTY1: Autoselect started
22:21:05: TTY1: Autoselect sample 7E
22:21:05: TTY1: Autoselect sample 7EFF
22:21:05: TTY1: Autoselect sample 7EFF7D
22:21:05: TTY1 Autoselect cmd: ppp default
22:21:05: TTY1: EXEC creation
%LINK-3-UPDOWN: Interface Async1, changed state to up
22:21:07: ppp: sending CONFREQ, type = 2 (CI_ASYNCMAP), value = A0000
22:21:07: ppp: sending CONFREQ, type = 5 (CI_MAGICNUMBER), value = 23BE13AA
22:21:08: PPP Async1: state = REQSENT fsm_rconfack(0xC021): rcvd id 0x11
22:21:08: ppp: config ACK received, type = 2 (CI_ASYNCMAP), value = A0000
22:21:08: ppp: config ACK received, type = 5 (CI_MAGICNUMBER), value = 23BE13AA
22:21:08: ppp: config ACK received, type = 7 (CI_PCOMPRESSION)
22:21:08: ppp: config ACK received, type = 8 (CI_ACCOMPRESSION)
22:21:08: PPP Async1: received config for type = 0x2 (ASYNCMAP) value = 0x0 acked
22:21:08: PPP Async1: received config for type = 0x5 (MAGICNUMBER) value = 0x2A acked
22:21:08: PPP Async1: received config for type = 0x7 (PCOMPRESSION) acked
22:21:08: PPP Async1: received config for type = 0x8 (ACCOMPRESSION) acked
```

```
22:21:08: ipcp: sending CONFREQ, type = 3 (CI_ADDRESS), Address = 172.16.1.1
22:21:08: ppp Async1: ipcp_reqci: rcvd COMPRESSTYPE (rejected) (REJ)
22:21:08: ppp Async1: Negotiate IP address: her address 0.0.0.0 (NAK with address
172.16.1.100) (NAK)
22:21:08: ppp: ipcp_reqci: returning CONFREJ.
22:21:08: PPP Async1: state = REQSENT fsm_rconfack(0x8021): rcvd id 0x9
22:21:08: ipcp: config ACK received, type = 3 (CI_ADDRESS), Address = 172.16.1.1
22:21:08: ppp Async1: Negotiate IP address: her address 0.0.0.0 (NAK with address
172.16.1.100) (NAK)
22:21:08: ppp: ipcp_reqci: returning CONFNAK.
22:21:09: ppp Async1: Negotiate IP address: her address 172.16.1.100 (ACK)
22:21:09: ppp: ipcp_reqci: returning CONFACK.
%LINEPROTO-5-UPDOWN: Line protocol on Interface Async1, changed state to up
```

Verify Autoselect ARA

The following trace appears when the **debug modem** command and **debug arap internal** command are enabled. As ARA version 2.0 calls pass through the access server, this output is displayed.

```
20:45:11: TTY3: DSR came up
20:45:11: tty3: Modem: IDLE->READY
20:45:11: TTY3: EXEC creation
20:45:11: TTY3: Autoselect(2) sample 1
20:45:11: TTY3: Autoselect(2) sample 11B
20:45:12: TTY3: Autoselect(2) sample 11B02
20:45:18: ARAP: ---------- SRVRVERSION ----------
20:45:19: ARAP: ---------- ACKing 0 ----------
20:45:19: ARAP: ---------- AUTH_CHALLENGE ----------
20:45:21: ARAP: ---------- ACKing 1 ----------
20:45:21: ARAP: ---------- AUTH_RESPONSE ----------
20:45:21: ARAP: ---------- STARTINFOFROMSERVER ----------
20:45:22: ARAP: ---------- ACKing 2 ----------
22:45:22: ARAP: ---------- ZONELISTINFO ----------
22:45:22: ARAP: ---------- ZONELISTINFO ----------
22:45:22: ARAP: ---------- ZONELISTINFO ----------
```

The following trace is for ARA version 1.0 calls.

```
22:31:45: TTY1: DSR came up
22:31:45: tty1: Modem: IDLE->READY
22:31:45: TTY1: Autoselect started
22:31:46: TTY1: Autoselect sample 16
22:31:46: TTY1: Autoselect sample 1610
22:31:46: TTY1: Autoselect sample 161002
22:31:47: ARAP: ---------- SRVRVERSION ----------
22:31:47: ARAP: ---------- ACKing 0 ----------
22:31:47: ARAP: ---------- AUTH_CHALLENGE ----------
22:31:47: ARAP: ---------- ACKing 1 ----------
22:31:47: ARAP: ---------- AUTH_RESPONSE ----------
22:31:47: ARAP: ---------- STARTINFOFROMSERVER ----------
22:31:48: ARAP: ---------- ACKing 2 ----------
22:31:48: ARAP: ---------- ZONELISTINFO ----------
22:31:48: ARAP: ---------- ZONELISTINFO ----------
22:31:49: ARAP: ---------- ZONELISTINFO ----------
```

Define a Command String for Automatic Execution

You can set up a command to execute automatically when the router connects to another host. The Cisco IOS software can execute any appropriate EXEC command and any switch or host name that occurs with the EXEC command. To define a command, use the following command in line configuration mode:

Command	Purpose
autocommand *command*	Define a command to be automatically executed.

Configure Rotary Groups

Connections can be made to the next free line in a group of lines, also called a *rotary group* or *hunt group*. A line can be in only one rotary group; a rotary group can consist of a single line or several contiguous lines. The console line (line 0) cannot be in a rotary group.

To configure a rotary group, use the following command in line configuration mode:

Command	Purpose
rotary *group*	Add a line to the specified rotary group.

Specify Decimal TCP Port Numbers when Connecting to Lines

Connections to an individual line are most useful when a dial-out modem, parallel printer, or serial printer is attached to that line. To connect to an individual line, the remote host or terminal must specify a particular TCP port on the router.

If reverse XRemote is required, that port is 9000 (decimal) plus the decimal value of the line number.

If a raw TCP stream is required, the port is 4000 (decimal) plus the decimal line number. The raw TCP stream is usually the required mode for sending data to a printer.

If Telnet protocols are required, that port is 2000 (decimal) plus the decimal value of the line number. The Telnet protocol might require that Return characters be translated into Return and line-feed character pairs. You can turn off this translation by specifying the Telnet binary mode option. To specify this option, connect to port 6000 (decimal) plus the decimal line number.

For example, a laser printer is attached to line 10 of a Cisco 2511 router. Such a printer usually uses XON/XOFF software flow control. Because the Cisco IOS software cannot receive an incoming connection if the line already has a process, you must ensure that an EXEC session is not accidentally started. You must, therefore, configure it as follows:

```
line 10
 flowcontrol software
 no exec
```

A host that wants to send data to the printer would connect to the router on TCP port 4008, send the data, and then close the connection. (Remember that line number 10 octal equals 8 decimal.)

Optimize Available Bandwidth

Asynchronous lines have relatively low bandwidth and can easily be overloaded, resulting in slow traffic across these lines.

To optimize available bandwidth, perform any of the following tasks:

- Configure Header Compression
- Force Header Compression at the EXEC Level

Configure Header Compression

One way to optimize available bandwidth is by using TCP header compression. Van Jacobson TCP header compression (defined by RFC 1144) can increase bandwidth availability between two and five times when compared to lines not using header compression. Theoretically, it can improve bandwidth availability by a ratio of seven to one.

To configure header compression, use the following command in interface configuration mode:

Command	Purpose
ip tcp header-compression [**on** | **off** | **passive**]	Configure Van Jacobson TCP header compression on the asynchronous link.

Force Header Compression at the EXEC Level

On SLIP interfaces, you can force header compression at the EXEC prompt on a line on which header compression has been set to passive. This allows more efficient use of the available bandwidth and does not require entering privileged configuration mode.

To implement header compression, use the following command in interface configuration mode:

Command	Purpose
ip tcp header-compression passive	Allow status of header compression to be assigned at the user level.

For PPP interfaces, the **passive** option functions the same as the **on** option.

Use Chat Scripts

Refer to the following sections to create and use chat scripts:

- Description
- Create a Chat Script
- Suggested Chat Script Naming Conventions (for Dial Scripts only)
- Configure the Line to Activate Chat Scripts
- Start a Chat Script Manually on an Asynchronous Line

Description

Chat scripts are strings of text used to send commands for modem dialing, logging onto remote systems, and initializing asynchronous devices connected to an asynchronous line. On a router, chat scripts can be configured on the auxiliary port only. A chat script must be configured to dial out on

asynchronous lines. You also can configure chat scripts so that they are executed automatically for other specific events on a line, or so that they are executed manually. Each chat script is defined for a different event. These events can include the following:

- Line activation
- Incoming connection initiation
- Asynchronous dial-on-demand routing
- Line resets
- Startup

The following tasks are required to use a chat script:

- Define the chat script in global configuration mode using the **chat-script** command.
- Configure the line so that a chat script is activated when a specific event occurs (using the **script** line configuration command), or start a chat script manually (using the **start-chat** privileged EXEC command).

Create a Chat Script

To define a chat script, use the following command in global configuration mode:

Command	Purpose
chat-script *script-name expect send...*	Create a script that will place a call on a modem, log on to a remote system, or initialize an asynchronous device on a line.

A limited list of keywords are supported, along with expect/send pairs. Send strings can have special escape modifiers.

Cisco recommends that one chat script (a "modem" chat script) be written for placing a call and another chat script (a "system" or "login" chat script) be written to log onto remote systems, where required.

Chat scripts are not supported on lines where modem control is set for inbound activity that only uses the **modem dialin** command.

Suggested Chat Script Naming Conventions (for Dial Scripts only)

When you create a script name, include the modem vendor, type, and modulation, separated by hyphens. For example, if you have a Telebit t3000 modem that uses V.32bis modulation, your script name would be *telebit-t3000-v32bis.*

A suggested naming convention for chat scripts used to dial is as follows:

vendor-type-modulation

In other words, the syntax of the **chat-script** command becomes the following:

chat-script *vendor-type-modulation expect send...*

For example, if you have a Telebit t3000 modem that uses V.32bis modulation, you would name your chat script as follows:

```
telebit-t3000-v32bis
```

The chat-script command could become the following:

```
router(config)# chat-script telebit-t3000-v32bis ABORT ERROR ABORT BUSY ABORT
"NO ANSWER" "" "ATH" OK "ATDT\T" TIMEOUT 30 CONNECT
```

Adhering to this naming convention allows you to specify a range of chat scripts using partial chat script names with regular expressions. This is particularly useful for dialer rotary groups.

Configure the Line to Activate Chat Scripts

Chat scripts can be activated by any of five events, each corresponding to a different version of the **script** line configuration command. To start a chat script manually at any point, refer to the following section, "Start a Chat Script Manually on an Asynchronous Line."

To define a chat script to start automatically when a specific event occurs, use the following commands in line configuration mode:

Step	Command	Purpose
1	**script activation** *regexp*[1]	Start a chat script on a line when the line is activated (every time a command EXEC is started on the line).
2	**script connection** *regexp*	Start a chat script on a line when a network connection is made to the line.
3	**script dialer** *regexp*	Specify a modem script for DDR on a line.
4	**script reset** *regexp*	Start a chat script on a line whenever the line is reset.
5	**script startup** *regexp*	Start a chat script on a line whenever the system is started up.

1 The argument *regexp* is a regular expression that is matched to a script name that has already been defined using the **chat-script** command.

Note Outbound chat scripts are not supported on lines where modem control is set for inbound activity only (using the **modem dialin** command).

Start a Chat Script Manually on an Asynchronous Line

You can start a chat script manually on any line that is currently not active by using the following command in privileged EXEC mode:

Command	Purpose
start-chat *regexp* [*line-number* [*dialer-string*]]	Start a chat script manually on any asynchronous line.

If you do not specify the line number, the script runs on the current line. If the line specified is already in use, you cannot start the chat script. A message appears indicating that the line is already in use.

Configuration Examples

This section illustrates different communication requirements for the following asynchronous scenarios:

- Restricting Access on the Asynchronous Interface Example
- Address Pooling on Asynchronous Interfaces Examples

- Group and Member Asynchronous Interfaces Examples
- Dedicated Asynchronous Interface Configuration Example
- IP and SLIP Using an Asynchronous Interface Example
- AppleTalk and PPP Example
- IP and PPP Example
- IPX and PPP Using a Loopback Interface Example
- IPX and PPP Using Dedicated IPX Network Numbers for Each Interface Example
- IPX and PPP over X.25 to an IPX Network on VTY Lines Example
- Remote Node NetBEUI Example
- Asynchronous Routing and Dynamic Addressing Configuration Example
- TCP Header Compression Configuration Example
- Conserving Network Addresses Using the IP Unnumbered Feature Example
- Configuring an Asynchronous Interface as the Only Network Interface Example
- Configuring Routing on a Dedicated Dial-In Router Example
- Configuring IGRP Example
- Configuring an Interface Example
- Remote Network Access Using PPP Basic Configuration Example
- Remote Network Access Using PPP and Routing IP Example
- Remote Network Access Using a Leased Line with Dial-Backup and PPP Example
- AT Mode Example for Integrated Modems

Restricting Access on the Asynchronous Interface Example

The following example assumes that users are restricted to certain servers designated as asynchronous servers, but that normal terminal users can access anything on the local network.

```
! access list for normal connections
access-list 1 permit 131.108.0.0 0.0.255.255
!
access-list 2 permit 131.108.42.55
access-list 2 permit 131.108.111.1
access-list 2 permit 131.108.55.99
!
line 1
 speed 19200
 flow hardware
 modem inout
interface async 1
 async mode interactive
 async dynamic address
 ip access-group 1 out
 ip access-group 2 in
```

Address Pooling on Asynchronous Interfaces Examples

The following sections provide examples of the use of DHCP and local pooling mechanisms.

DHCP Pooling Examples

The following global configuration example enables DHCP proxy-client status on all asynchronous interfaces on the access server:

```
ip address-pool dhcp-proxy-client
```

The following global configuration example illustrates how to specify which DHCP servers are used on your network. You can specify up to four servers using IP addresses or names. If you do not specify servers, the default is to use the IP limited broadcast address of 255.255.255.255 for transactions with any and all discovered DHCP servers.

```
ip dhcp-server jones smith wesson
```

The following interface configuration example illustrates how to disable DHCP proxy-client functionality on asynchronous interface 1:

```
async interface
interface 1
 no peer default ip address
```

Local Pooling Example

The following example shows how to select the IP pooling mechanism and how to create a pool of local IP addresses that are used when a client dials in on an asynchronous line. The default address pool comprises IP addresses 172.30.0.1 through 172.30.0.28.

```
! this command tells the access server to use a local pool
ip address-pool local
! this command defines the ip address pool.
! The address pool is named group1 and comprised of addresses
! 10.1.2.1through 10.1.2.5 inclusive
ip local-pool group1 10.1.2.1 10.1.2.5
```

Configure Specific IP Addresses for an Interface Example

This example shows how to configure the access server so that it will use the default address pool on all interfaces except interface 7, on which it will use an address pool called lass:

```
ip address-pool local
ip local-pool lass 172.30.0.1
 async interface
 interface 7
 peer default ip address lass
```

Group and Member Asynchronous Interfaces Examples

The following example shows how to create an asynchronous group interface 0 with group interface members 2 through 7, starting in global configuration mode:

```
interface group-async 0
 group-range 2 7
```

The following example shows how you need to configure asynchronous interfaces 1, 2, and 3 separately if you do not have a group interface configured:

```
interface Async1
 ip unnumbered Ethernet0
 encapsulation ppp
 async default ip address 172.30.1.1
 async mode interactive
 async dynamic routing
!
interface Async2
 ip unnumbered Ethernet0
 encapsulation ppp
 async default ip address 172.30.1.2
 async mode interactive
 async dynamic routing
!
interface Async3
 ip unnumbered Ethernet0
!
 encapsulation ppp
 async default ip address 172.30.1.3
 async mode interactive
 async dynamic routing
```

The following example configures the same interfaces, but from a single group asynchronous interface:

```
!
interface Group-Async 0
 ip unnumbered Ethernet0
 encapsulation ppp
 async mode interactive
 async dynamic routing
 group-range 1 3
 member 1 async default ip address 172.30.1.1
 member 2 async default ip address 172.30.1.2
 member 3 async default ip address 172.30.1.3
```

Dedicated Asynchronous Interface Configuration Example

The following example assigns an IP address to an asynchronous interface and places the line in dedicated network mode. Setting the stop bit to 1 is a performance enhancement.

```
line 20
 location Department PC Lab
 stopbits 1
 speed 19200
!
interface async 20
 async default ip address 182.32.7.51
 async mode dedicated
```

IP and SLIP Using an Asynchronous Interface Example

The following example configures IP–SLIP on asynchronous interface 6. The IP address for the interface is assigned to Ethernet 0, interactive mode has been enabled, and the IP address of the client PC running SLIP has been specified.

IP and the appropriate IP routing protocols have already been enabled on the access server or router.

```
interface async 6
 ip unnumbered ethernet 0
 encapsulation slip
 async mode interactive
 async default ip address 172.18.1.128
```

AppleTalk and PPP Example

The following example configures asynchronous interface 4 on the router so that users can access AppleTalk zones by dialing into the router via PPP to this interface. Users accessing the network can run AppleTalk and IP natively on a remote Macintosh, access any available AppleTalk zones from Chooser, use networked peripherals, and share files with other Macintosh users. Routing is not supported on the asynchronous interface 6.

```
interface async 6
 encapsulation ppp
 appletalk virtual-net 12345 saivite
 appletalk client-mode
```

IP and PPP Example

The following example configures IP–PPP on asynchronous interface 6. The IP address for the interface is assigned to Ethernet 0, interactive mode has been enabled, and the IP address of the client PC running PPP has been specified.

IP and the appropriate IP routing protocols have already been enabled on the access server or router.

```
interface async 6
 ip unnumbered ethernet 0
 encapsulation ppp
 async mode interactive
 peer default ip address 172.18.1.128
```

IPX and PPP Using a Loopback Interface Example

The following example shows how to configure IPX to run over PPP on an asynchronous interface. The asynchronous interface is associated with a loopback interface configured to run IPX. This example enables a non-routing IPX client to connect to the router.

```
ipx routing 0000.0c07.b509
interface loopback0
 no ip address
 ipx network 544
 ipx sap-interval 2000
interface ethernet0
 ip address 172.21.14.64
 ipx network AC150E00
 ipx encapsulation SAP
interface async 3
 ip unnumbered ethernet0
 encapsulation ppp
 async mode interactive
 async default ip address 172.18.1.128
 ipx ppp-client loopback0
 ipx sap-interval 0
```

In this example, IPX client connections are permitted to asynchronous interface 3, which is associated with loopback interface 0. Loopback interface 0 is configured to run IPX. Routing updates have been filtered on asynchronous interface 3. Routing updates take up a great deal of bandwidth, and asynchronous interfaces have low bandwidth.

IPX and PPP Using Dedicated IPX Network Numbers for Each Interface Example

The following example shows how to configure IPX to run over PPP on an asynchronous interface. A dedicated IPX network number has been specified for each interface, which can require a substantial number of network numbers for a large number of interfaces. This example permits an IPX client with routing enabled to connect with the router.

```
ipx routing 0000.0c07.b509
interface async 6
 ip unnumbered ethernet0
 encapsulation ppp
 async mode interactive
 ipx network AC150E00
 ipx sap-interval 0
```

In this example, IPX client connections are permitted to asynchronous interface 6, which has a unique IPX network number. Routing updates have been filtered on asynchronous interface 6. Routing updates take up a great deal of bandwidth, and asynchronous interfaces have low bandwidth.

IPX and PPP over X.25 to an IPX Network on VTY Lines Example

The following example shows how to enable IPX–PPP on VTY lines. First, you enable PPP to run on VTY lines, then you associate the VTY line with a loopback interface configured to run IPX. This example enables a non-routing IPX client to connect to the router.

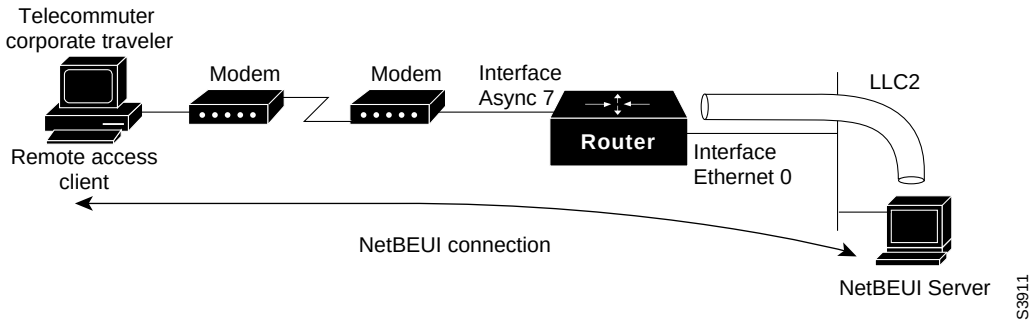
```
ipx routing 0000.0c07.b509
interface loopback0
 no ip address
 ipx network 544
vty-async ipx ppp-client loopback0
```

In this example, IPX client connections are permitted to VTY lines, which have been associated with loopback interface 0. Loopback interface 0 is configured with an IPX network number that is used by the VTY lines.

Remote Node NetBEUI Example

In the following example, asynchronous interface 7 and ethernet interface 0 are configured to enable NetBEUI connectivity between the corporate telecommuter's client and the remote access (NetBEUI) server. The PC client is running a legacy application—Chat—in Windows NT, to connect with the remote server. Refer to Figure 25.

Figure 25 Connecting a Remote NetBEUI Client to a Server through a Router



The configuration for the router is as follows:

```
interface async 7
 netbios nbf
 encapsulation ppp
```

You would also need to configure security, such as TACACS+, RADIUS, or another form of login authentication on the router.

Asynchronous Routing and Dynamic Addressing Configuration Example

The following example shows a simple configuration that allows routing and dynamic addressing. With this configuration, if the user specifies **/routing** in the EXEC **slip** or **ppp** command, routing protocols will be sent and received.

```
interface async 6
 async dynamic routing
 async dynamic address
```

TCP Header Compression Configuration Example

The following example configures async interface 7 with a default IP address, allowing header compression if it is specified in the **slip** or **ppp** connection command entered by the user or if the connecting system sends compressed packets.

```
interface async 7
 ip address 172.31.79.1
 async default ip address 172.31.79.2
 ip tcp header-compression passive
```

Conserving Network Addresses Using the IP Unnumbered Feature Example

The following example shows how to configure your router for routing using unnumbered interfaces. The source (local) address is shared between Ethernet 0 and async 6 (172.18.1.1). The default remote address is 172.18.1.2.

```
interface ethernet 0
 ip address 172.18.1.1 255.255.255.0
!
interface async 6
 ip unnumbered ethernet 0
 async dynamic routing
! default address is on the local subnet
 async dynamic address
 async default ip address 172.18.1.2
 ip tcp header-compression passive
```

The following example shows how the IP unnumbered configuration works. Although the user assigned an address, the system response shows the interface as unnumbered, and the address entered by the user will be used only in response to BOOTP requests.

```
router> slip /compressed 10.11.11.254
Password:
Entering async mode.
Interface IP address is unnumbered, MTU is 1500 bytes.
Header compression is On.
```

Configuring an Asynchronous Interface as the Only Network Interface Example

In the following example, one of the asynchronous lines is used as the only network interface. The router is used primarily as a terminal server, but is at a remote location and dials into the central site for its only network connection.

```
ip default-gateway 10.11.12.2
interface ethernet 0
 shutdown
interface async 1
 async dynamic routing
 ip tcp header-compression on
 async default ip address 10.11.16.12
 async mode dedicated
 ip address 10.11.12.32 255.255.255.0
```

Configuring Routing on a Dedicated Dial-In Router Example

In the following example, the router is set up as a dedicated dial-in router. Interfaces are configured as IP unnumbered to conserve network resources, primarily IP addresses.

```
ip routing
interface ether 0
 ip address 10.129.128.2 255.255.255.0
!
interface async 1
 ip unnumbered ethernet 0
 async dynamic routing
! The addresses assigned with SLIP or PPP EXEC commands are not used except
! to reply to BOOTP requests.
! Normally, the routers dialing in will have their own address
! and not use BOOTP at all.
 async default ip address 10.11.11.254
!
interface async 2
 ip unnumbered ethernet 0
 async default ip address 10.11.12.16
 ip tcp header-compression passive
 async mode dedicated
!
! run RIP on the asynchronous lines, because few implementations of SLIP
! understand IGRP. Run IGRP on the ethernet (and in the local network).
!
router igrp 110
 network 10.11.12.0
! send routes from the asynchronous lines on the production network.
 redistribute RIP
! don't send IGRP updates on the async interfaces
 passive-interface async 1
!
 router RIP
 network 10.11.12.0
 redistribute igrp
 passive-interface ethernet 0
! consider filtering everything except a default route from the routing
! updates sent on the (slow) asynchronous lines
 distribute-list 1 out
 ip unnumbered async 2
 async dynamic routing
```

Configuring IGRP Example

In the following example, only the IGRP TCP/IP routing protocol is running; it is assumed that the systems that are dialing in to use routing will either support IGRP or have some other method (for example, a static default route) of determining that the router is the best place to send most of its packets.

```
router igrp 111
 network 10.11.12.0
interface ethernet 0
 ip address 10.11.12.92 255.255.255.0
!
interface async 1
 async default ip address 10.11.12.96
 async dynamic routing
 ip tcp header-compression passive
 ip unnumbered ethernet 0

line 1
 modem ri-is-cd
```

Configuring an Interface Example

The following configuration shows interface and line configuration. The interface is configured with access lists, passive header compression and a default address. The line is configured for TACACS authentication.

```
interface async 1
 ip access-group 1 in
 ip access-group 1 out
 ip tcp header-compression passive
 async default ip address 172.31.176.201

line 1
 login tacacs
 location 457-5xxx
 exec-timeout 20 0
 password XXXXXXXX
 session-timeout 20
 stopbits 1
```

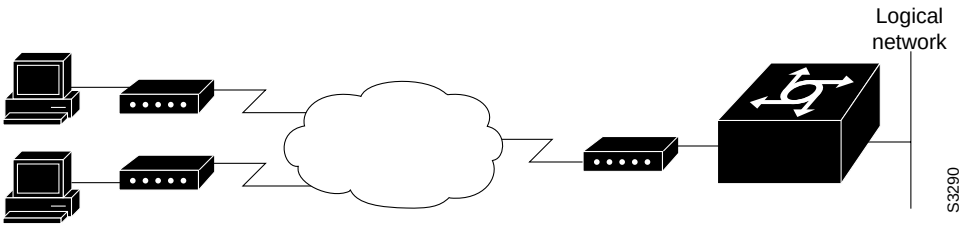
Remote Network Access Using PPP Basic Configuration Example

Figure 26 illustrates a simple network configuration comprised of remote PCs with modems connected via modem to a router. The cloud is a public switched telephone network (PSTN). The modems are connected via asynchronous lines, and the access server is connected to a local network.

In this configuration you need to configure the following:

- An asynchronous line on the access server configured to use PPP encapsulation
- An interface on the access server for the modem connection; this interface also needs to be configured to accept incoming modem calls
- A default IP address for each incoming line

Figure 26 Remote Network Access Using PPP



This default address indicates the address of the remote PC to the server, unless the user explicitly specifies another when starting the PPP session.

The server is configured for interactive mode with autoselect enabled, which allows the user to automatically begin a PPP session upon detection of a PPP packet from the remote PC; or, the remote PC can explicitly begin a PPP session by typing PPP at the prompt.

The configuration is as follows:

```
ip routing
!
interface ethernet 0
 ip address 192.168.32.12 255.255.255.0
!
interface async 1
 encapsulation ppp
 async mode interactive
 async default ip address 192.168.32.51
 async dynamic address
 ip unnumbered ethernet 0

line 1
 autoselect ppp
 modem callin
 speed 19200
```

Remote Network Access Using PPP and Routing IP Example

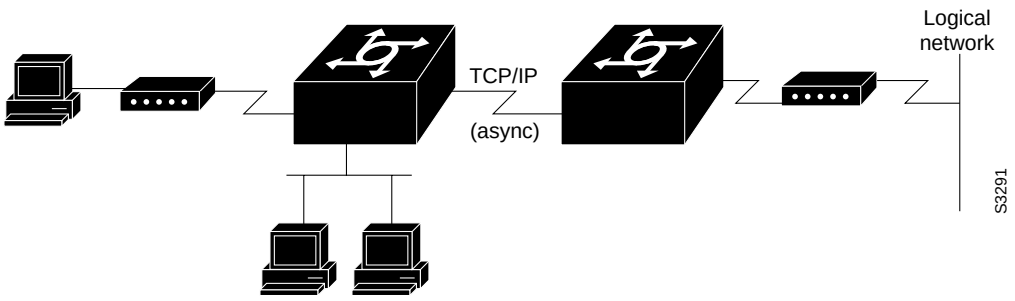
Figure 27 illustrates a network configuration that provides routing functionality, allowing routing updates to be passed across the asynchronous lines.

This network is comprised of remote and local PCs connected via modem and network connections to an access server. This access server is connected to a second access server via an asynchronous line running TCP/IP. The second access server is connected to a local network via modem.

For this scenario, you will need to configure the following:

- An asynchronous line on both access servers configured to use PPP encapsulation
- An interface on both access servers for the modem connection and for this interface to be configured to accept incoming modem calls
- A default IP address for each incoming line
- IP routing on all configured interfaces

Figure 27 Routing on an Asynchronous Line Using PPP



The configuration is as follows:

```
interface async 1
 encapsulation ppp
 async mode interactive
 async default ip address 192.168.32.10
 async dynamic address
 ip unnumbered ethernet 0
 async dynamic routing
```

If you want to pass IP routing updates across the asynchronous link, issue the following commands:

```
line 1
 autoselect ppp
 modem callin
 speed 19200
```

Next, enter these commands to configure the asynchronous lines between the access servers, starting in global configuration mode:

```
interface async 2
 async default ip address 192.168.32.55
 ip tcp header compression passive
```

Finally, configure routing as described in the *Network Protocols Configuration Guide, Part 1* using one of the following methods. The server can route packets three different ways:

1. Use ARP, which is the default behavior.
2. Use a default-gateway by issuing the command **ip default-gateway** *x.x.x.x*, where *x.x.x.x* is the IP address of a locally attached router.
3. Run an IP routing protocol (RIP, IGRP, EIGRP, or OSPF).

Remote Network Access Using a Leased Line with Dial-Backup and PPP Example

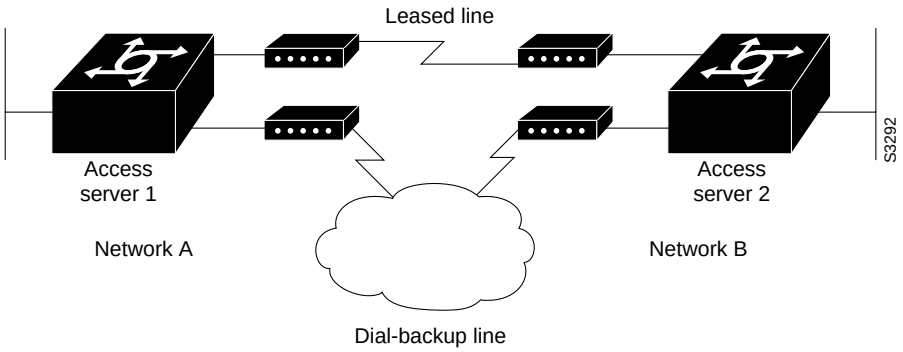
Figure 28 illustrates a scenario where two networks are connected via access servers on a leased line. Redundancy is provided by a dial-backup line over the public switched telephone network so that if the primary leased line goes down, the dial-backup line will be automatically brought up to restore the connection. This configuration would be useful for using an auxiliary port as the backup port for a synchronous port.

In this scenario, you will need to configure the following:

- Two asynchronous interfaces on each access server
- Two modem interfaces

- A default IP address for each interface
- Dial-backup on one modem interface per access server
- An interface connecting to the access server's related network

Figure 28 Asynchronous Leased Line with Backup



The configuration is as follows:

```
hostname routerA
!
username routerB password cisco
chat-script backup "" "AT" TIMEOUT 30 OK atdt\T TIMEOUT 30 CONNECT \c !
!
interface Serial0
 backup interface Async1
 ip address 192.168.222.12 255.255.255.0
!
interface Async1
 ip address 172.16.199.1 255.255.255.0
 encapsulation ppp
 async default ip address 172.16.199.2
 async dynamic address
 async dynamic routing
 async mode dedicated
 dialer in-band
 dialer map IP 172.16.199.2 name routerB modem-script backup broadcast 3241129
 dialer-group 1
 ppp authentication chap
!
 dialer-list 1 protocol ip permit
!
line aux 0
 modem InOut
 rxspeed 38400
 txspeed 38400
```

AT Mode Example for Integrated Modems

To establish a direct connect session to an internal or integrated modem (existing inside the router), such as required for Microcom modems in the Cisco AS5200, first open a directly connected session with the **modem at-mode** command then send an AT command to the specified modem. For example, the following example sends the AT command **at\s** to modem 1/1:

```
AS5200# modem at-mode 1/1
You are now entering AT command mode on modem (slot 1 / port 1).
Please type CTRL-C to exit AT command mode.
at%v

MNP Class 10 V.34/V.FC Modem Rev 1.0/85

OK
at\s

IDLE           000:00:00
LAST DIAL

NET ADDR:      FFFFFFFFFFFF
MODEM HW: SA 2W United States
4 RTS 5 CTS 6 DSR - CD 20 DTR - RI
MODULATION     IDLE
MODEM BPS      28800  AT%G0
MODEM FLOW     OFF    AT\G0
MODEM MODE     AUT    AT\N3
V.23 OPR.      OFF    AT%F0
AUTO ANS.      ON     ATS0=1
SERIAL BPS     115200 AT%U0
BPS ADJUST     OFF    AT\J0
SPT BPS ADJ.   0      AT\W0
ANSWER MESSGS  ON     ATQ0
SERIAL FLOW    BHW    AT\Q3
PASS XON/XOFF  OFF    AT\X0
PARITY         8N     AT
```