Configure Tcl IVR Applications

Overview

This chapter shows you how to configure Cisco Unified IP Interactive Voice Response (IVR) using the Tool Command Language (TCL) scripts. This chapter contains the following sections:

To identify the hardware platform or software image information that is associated with a feature in this chapter, use the Feature Navigator on Cisco.com to search for information about the feature or refer to the software release notes for a specific release.

IVR consists of simple voice prompting and digit collection to gather caller information for authenticating the user and identifying the destination. It is possible to assign IVR applications to specific ports or invoke on the basis of DNIS. An IP public switched telephone network gateway can have several IVR applications to accommodate many different gateway services, and you can customize the IVR applications to present different interfaces to the various callers.

IVR systems provide information in the form of recorded messages over phone lines in response to user input in the form of spoken words, or more commonly dual tone multifrequency (DTMF) signaling. For example, when a user makes a call with a debit card, an IVR application is used to prompt the caller to enter a specific type of information, such as an account number. After playing the voice prompt, the IVR application collects the predetermined number of touch tones and then places the call to the destination phone or system.

IVR uses TCL scripts gather information and to process accounting and billing. For example, a TCL IVR script plays when a caller receives a voice-prompt instruction to enter a specific type of information, such as a Personal Identification Number (PIN). After playing the voice prompt, the TCL IVR application collects the predetermined number of touch tones and sends the collected information to an external server for user authentication and authorization.

| **Note** | H.323 protocol is no longer supported from Cisco IOS XE Bengaluru 17.6.1a onwards. Consider using SIP for multimedia applications. |

Tcl IVR Enhancements

Since the introduction of the Cisco IVR technology, the software has undergone several enhancements. TCL IVR Version 2.0 is made up of separate components that are described individually in the sections that follow. The enhancements are as follows:

- Real Time Streaming Protocol (RTSP) client implementation

- TCL IVR prompt playout and digit collection on IP call legs

- New TCL verbs to utilize RTSP scripting features.

The enhancements add scalability and enable the TCL IVR scripting functionality on VoIP legs. In addition, support for RTSP enables VoIP gateways to play messages from RTSP-compliant announcement servers. The addition of these enhancements also reduces the CPU load and saves memory on the gateway because no packetization is involved. Larger prompts can be played, and the use of an external audio server is allowed.

| **Note** | TCL IVR 2.0 removed the signature locking mechanism requirement. |

TCL IVR Prompts Played on IP Call Legs

TCL IVR Version 2.0 scripts is configured for incoming plain old telephone service (POTS) or VoIP call legs to play announcements to the user or collect user input (digits). With TCL IVR Version 2.0 the prompts is triggered from both the PSTN side of the call leg and the IP side of the call leg. This feature enables the audio files (or prompts) to be played out over the IP network.

TCL IVR scripts played toward a VoIP call leg are subject to the following conditions:

- G.711 mu-law encoding must be used when prompts are played.

- G.711 mu-law encoding must also be used while these calls, even at prompt playout has completed.

- Digital Signaling Protocols (DSPs) cannot be on the IP call leg so the script cannot initiate a tone.

- When a TCL IVR script is used to collect digits on a VoIP call leg, use Cisco proprietary RTP for SIP protocol configured on the call leg.

| **Note** | For additional information about the **dtmf-relay** command, refer to the Cisco IOS Voice Command Reference - D through I. |

IVR 2.0 enables the system to accept calls that are initiated from the IP side of the network using G.711, and terminate calls to the terminating gateway using the same codec. IP phones can also originate a call to a CUBE running an TCL IVR script.

TCL Verbs

TCL IVR, Version 2.0, delivers a new set of TCL verbs and scripts that replace the previous TCL version. The TCL verbs enable the user to develop TCL scripts that interact with the IVR application.

TCL IVR Version 2.0 is not backward compatible with the IVR 1.0 scripts.

TCL IVR scripts use the TCL verbs to interact with the gateway during call processing in order to collect the required digits—for example, to request the PIN or account number for the caller. The TCL scripts are the default scripts for all Cisco Voice features using IVR. TCL scripts are configured to control calls coming into or going out of the gateway.

Note Ensure that you have loaded the version of TCL scripts that support IVR Version 2.

The TCL IVR scripts that are shown below are listed as an example of the types of scripts available to be downloaded from the Cisco.com Software Center. For a complete list of scripts, it is recommended that you check the Software Center.

Cisco provides the following IVR scripts:

- fax_hop_on_1—Collects digits from the redialer, such as account number and destination number.

- clid_authen—Authenticates the call with Automatic Number Identification (ANI) and DNIS numbers, collects the destination data, and makes the call.

- clid_authen_npw—Performs as clid_authen, but uses a null password when authenticating, rather than DNIS numbers.

- clid_authen_collect—Authenticates the call with ANI and DNIS numbers and collects the destination data. If authentication fails, it collects the account and password.

- clid_authen_col_npw—Performs as clid_authen_collect, but uses a null password and does not use or collect DNIS numbers.

- clid_col_npw_3—Performs as clid_authen_col_npw except with that script, if authentication with the digits collected (account and PIN) fails, the clid_authen_col_npwscript just plays a failure message (auth_failed.au) and then hangs up. The clid_col_npw_3 script allows two failures, then plays the retry audio file (auth_retry.au) and collects the account and PIN again.

- The caller can interrupt the message by entering digits for the account number, triggering the prompt to tell the caller to enter the PIN. If authentication fails the third time, the script plays the audio file auth_fail_final.au, and hangs up.

The following table lists the prompt audio files that are associated with the clid_col_npw_3script.

Table 1: clid_col_npw_3 Script Prompt Audio Files

Audio Filename	Action
flash:enter_account.au	Asks the caller to enter an account number. Played as the first request.
flash:auth_fail_retry.au	Asks the caller to reenter the account number. Plays after two failures.

flash:enter_pin.au	Asks the caller to enter a PIN.
flash:enter_destination.au	Asks the caller to enter a destination phone number.
flash:auth_fail_final.au	Informs the caller that the account number authorization has failed three times.

The following table lists additional audio files that are associated with the clid_col_npw_3script.

Table 2: Additional clid_col_npw_3 Script Audio Files

Audio Filename	Action
auth_fail_retry.au	Informs the caller that authorization failed. Prompts the caller to reenter the account number followed by the pound sign (#).
auth_fail_final.au	Informs the caller, "I'm sorry, your account number cannot be verified. Please hang up and try again."

- clid_col_npw_npw—Tries to authenticate by using ANI, null as the user ID, user, and user password pair. If that fails, it collects an account number and authenticates with account and null. It allows three tries for the caller to enter the account number before ending the call with the authentication failed audio file. If authentication succeeds, it plays a prompt to enter the destination number.

The following table lists the audio files that are associated with the clid_col_npw_npw script.

Table 3: clid_col_npw_npw Script Audio Files

Audio Filename	Action
flash:enter_account.au	Asks the caller to enter the account number the first time.
flash:auth_fail_retry.au	Asks the caller to reenter the account number after first two failures.
flash:enter_destination.au	Asks the caller to enter the destination phone number.
flash:auth_fail_final.au	Informs the caller that the account number authorization has failed three times.

- clid_col_dnis_3.tcl—Authenticates the caller ID three times. First it authenticates the caller ID with DNIS. If that is not successful, it attempts to authenticate with the caller PIN up to three times.

- clid_col_npw_3.tcl—Authenticates with null. If authentication is not successful, it attempts to authenticate by using the caller PIN up to 3 times.

- clid_4digits_npw_3.tcl—Authenticates with null. If the authentication is not successful, it attempts to authenticate with the caller PIN up to 3 times using the 14-digit account number and password entered together.

- clid_4digits_npw_3_cli.tcl—Authenticates the account number and PIN respectively by using ANI and null. The number of digits that are allowed for the account number and password are configurable through

the CLI. If the authentication fails, it allows the caller to retry. The retry number is also configured through the CLI.

- clid_authen_col_npw_cli.tcl—Authenticates the account number and PIN respectively using ANI and null. If the authentication fails, it allows the caller to retry. The retry number is configured through the CLI. The account number and PIN are collected separately.

- clid_authen_collect_cli.tcl—Authenticates the account number and PIN by using ANI and DNIS. If the authentication fails, it allows the caller to retry. The retry number is configured through the CLI. The account number and PIN are collected separately.

- clid_col_npw_3_cli.tcl—Authenticates by using ANI and null for account and PIN respectively. If the authentication fails, it allows the caller to retry. The retry number is configured through the CLI.

- clid_col_npw_npw_cli.tcl—Authenticates by using ANI and null for account and PIN respectively. If authentication fails, it allows the caller to retry. The retry number is configured through the CLI. The account number and PIN are collected together.

Note To display the contents of the TCL IVR script, use the show call application voice command.

Prerequisites

Before you configure your CUBE to support TCL IVR, you must perform the following prerequisite tasks:

- Configure VoIP to support SIP—meaning that in addition to the basic configuration tasks, such as configuring dial peers and voice ports, you must configure specific devices in your network to act as gateways.

- Configure a TFTP sever to perform storage and retrieval of the audio files, which are required by the Debit Card gateway or other features requiring TCL IVR scripts and audio files.

- Download the appropriate TCL IVR script from the https://d1nmyq4gcgsfi5.cloudfront.net/site/voice-gateway/downloads/sample-scripts/ . Use the **copy** command to copy your audio file (.au file) to your flash memory, and the **audio-prompt load** command to read it into RAM. When you use TCL IVR applications, the CUBE must know the URL where the TCL script can be found, and the URL of any audio file you want to use. Cisco IOS File System (IFS) is used to read the files, so any IFS-supported URLs can be used, which includes TFTP, FTP, or a pointer to a device on the router. During configuration of the application, you specify the URLs for the script and for the audio prompt. See the "Using URLs in IVR Scripts" chapter in the *TCL IVR API Version 2.0 Programmer's Guide* for more information.

- Make sure that your audio files are in the proper format. The TCL IVR prompts require audio file (.au) format of 8 bit, u-law, and 8-khz encoding.

- Install and configure the appropriate RADIUS security server in your network. The version of RADIUS that you are using must be able to support IETF-supported vendor specific attributes (VSAs), which are implemented by using IETF RADIUS attribute 26.

TCL IVR Configuration Tasks

Before starting the software configuration tasks for the TCL IVR Version 2.0 features, complete the following preinstallation tasks:

- Download the TCL scripts and audio files to be used with this feature from the https://d1nmyq4gcgsfi5.cloudfront.net/site/voice-gateway/downloads/sample-scripts/.

- Store the TCL scripts and audio files on a TFTP server that is accessible by CUBE.

- Create the TCL IVR application script to use with the **call application voice** command when configuring IVR using TCL scripts. You create this application first and store it on a server or location for the easy retrieval.

- Define the call flow and pass the defined parameter values to the application. Depending on the TCL script you select, these values can include the language of the audio file and the location of the audio file. Table 1: clid_col_npw_3 Script Prompt Audio Files, on page 3 Lists the TCL scripts and the parameter values they require.

- Associate the application to the incoming VoIP dial peer.

Configure the Call Application for the Dial Peer

Before you begin

You must configure the application that interacts with the dial peer before you configure the dial peer. The dial peer collects digits from the caller and uses the application that you have created. Use the call application voice command as shown in the table that follows. Each command line is optional depending on the type of action that is desired or the digits to be collected.

To configure the application, enter the following commands in global configuration mode:

SUMMARY STEPS

1. **call application voice** *name url*
2. **call application voice** *name* **language** *digit language*
3. **call application voice** *name* **pin-length** *number*
4. **call application voice** *name* **retry-count** *number*
5. **call application voice** *name* **uid-length** *number*
6. **call application voice** *name* **set-location** *language category location*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**call application voice** *name url* **Example:** `Router(config)# `**`call application voice`** `name url`	Defines the name of the application to be used with your TCL IVR script. The *url* argument specifies the location of the file and the access protocol. An example is as follows: `flash:scripts/session.tcl`

	Command or Action	Purpose
| | ```
tftp://dirt/sarvi/scripts/session.tcl
ftp://sarvi-ultra/scripts/session.tcl
slot0:scripts/tcl/session..tcl
``` **Note** You can only configure a url if the application named *name* has *not* been configured. |  |
Step 2	**call application voice** *name* **language** *digit language* **Example:** `Router(config)# call application voice name language digit language`	Specifies the language that is used by the audio files. An example is: call application voice test language 1 en. The arguments are as follows: • *digit*—Specifies zero (0) through 9. • *language*—Specifies two characters that represent a language. For example, "en" for English, "sp" for Spanish, and "ch" for Mandarin. Enter **aa** to represent all.
Step 3	**call application voice** *name* **pin-length** *number* **Example:** `Router(config)# call application voice name pin-length number`	Defines the number of characters in the PIN for the designated application. Values are from 0 through 10.
Step 4	**call application voice** *name* **retry-count** *number* **Example:** `Router(config)# call application voice name retry-count number`	Defines the number of times a caller is permitted to reenter the PIN for the designated application. Values are from 1 through 5.
Step 5	**call application voice** *name* **uid-length** *number* **Example:** `Router(config)# call application voice name uid-length number`	Defines the number of characters that are allowed to be entered for the user ID for the designated application. Values are from 1 through 20.
Step 6	**call application voice** *name* **set-location** *language category location* **Example:** `Router(config)# call application voice`*name*`set-location`*language category location*	Defines the location, language, and category of the audio files for the designated application. An example is: **set-location** en 1 tftp://server dir/audio filename.

What to do next

The following table lists TCL script names and the corresponding parameters that are required for each TCL script.

Table 4: TCL Scripts and Parameters

TCL Script Name	Description—Summary	Commands to Configure
clid_4digits_npw_3_cli.tcl	Authenticates the account number and PIN using ANI and null. The allowed length of digits is configurable through the CLI. If the authentication fails, it allows the caller to retry. The retry number is also configured through the CLI.	**call application voice uid-len** min = 1, max = 20, default - 10 **call application voice pin-len** min = 0, max - 10, default = 4 **call application voice retry-count** min = 1, max = 5, default = 3
clid_authen_col_npw_cli.tcl	Authenticates the account number and PIN using ANI and null. If the authentication fails, it allows the caller to retry. The retry number is configured through the CLI. The account number and PIN are collected separately.	**call application voice retry-count** min = 1, max = 5, default = 3
clid_authen_collect_cli.tcl	Authenticates the account number and PIN using ANI and DNIS. If the authentication fails, it allows the caller to retry. The retry number is configured through the CLI. The account number and PIN are collected separately.	**call application voice retry-count** min = 1, max = 5, default = 3
clid_col_npw_3_cli.tcl	Authenticates using ANI and null for account and PIN. If the authentication fails, it allows the caller to retry. The retry number is configured through the CLI.	**call application voice retry-count** min = 1, max = 5, default = 3
clid_col_npw_npw_cli.tcl	Authenticates using ANI and null for account and PIN. If authentication fails, it allows the caller to retry. The retry number is configured through the CLI. The account number and PIN are collected together.	**call application voice retry-count** min = 1, max = 5, default = 3

Configure TCL IVR on the Inbound VoIP Dial Peer

Before you begin

To configure the inbound VoIP dial peer, use the following commands beginning in global configuration mode:

SUMMARY STEPS

1. **dial-peer voice** *4401 voip*
2. **gw-accounting VOIP**
3. **aaa authentication login VOIP radius**
4. **aaa accounting connection VOIP start-stop radius**
5. **radius-server host** *ip-address* **auth-port** *number* **acct-port** *number*
6. **application** *application-name*
7. **destination-pattern** *pattern*
8. **session protocol** *sipv2*
9. **session target**
10. **dtmf-relay rtp-nte**
11. **codec g711ulaw**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**dial-peer voice** *4401 voip* **Example:** Router(config)# **dial-peer voice** *4401 voip*	Enters the dial-peer configuration mode and identifies the call leg.
Step 2	**gw-accounting VOIP** **Example:** Router(config)# **gw-accounting h323**	(Optional) Enables gateway-specific VOIP accounting.
Step 3	**aaa authentication login VOIP radius** **Example:** Router(config)# **aaa authentication login VOIP radius**	(Optional) Defines a method list that is called VOIP where RADIUS is defined as the only method of login authentication.
Step 4	**aaa accounting connection VOIP start-stop radius** **Example:** Router(config)# **aaa accounting connection VOIP start-stop radius**	(Optional) Defines a method list that is called VOIP where RADIUS is used to perform connection accounting, providing start-stop records.
Step 5	**radius-server host** *ip-address* **auth-port** *number* **acct-port** *number* **Example:** Router(config)# **radius-server host** *ip-address auth-port number acct-port number*	Identifies the RADIUS server and the ports that will be used for authentication and accounting services.
Step 6	**application** *application-name* **Example:** Router(config-dial-peer)# **application** *application-name*	Specifies the name of the application and script to use.

	Command or Action	Purpose
Step 7	**destination-pattern** *pattern* **Example:** `Router(config-dial-peer)# destination-pattern pattern`	Enters the destination pattern.
Step 8	**session protocol** *sipv2* **Example:** `Router(config-dial-peer)# session protocol sipv2`	Specifies the session protocol. The default session protocol is VOIP. The *sipv2* argument enables SIP.
Step 9	**session target** **Example:** `Router(config-dial-peer)# session target`	Specifies the session target IP address.
Step 10	**dtmf-relay rtp-nte** **Example:** `Router(config-dial-peer)# dtmf-relay rtp-nte`	Specifies the DTMF relay method. The keyword **rtp-nte** specifies VOIP and SIP. **Note** If digit collection from this VoIP call leg is required, the command **dtmf-relay** is required. The default is **no dtmf-relay**.
Step 11	**codec g711ulaw** **Example:** `Router(config-dial-peer)# codec g711ulaw`	Specifies the voice codec. **Note** If the configured application is playing prompts to the VoIP call leg, the **g711ulaw** keyword is required.

Verify TCL IVR Configuration

Before you begin

You can verify TCL IVR configuration by performing the following tasks:

- To verify TCL IVR configuration parameters, use the show running-config command.
- To display a list of all voice applications, use the show call application summary command.
- To display a list of all voice applications, use the show call application summary command.
- To show the contents of the script configured, use the show call application voice command.
- To verify that the operational status of the dial peer, use the show dial-peer voice command.

To verify the TCL IVR configuration, perform the following steps:

Step 1   Enter the show call application voice summary command to verify that the newly created applications are listed. The example output follows

```
Router# show call application voice summary
```

Name	Description
DEFAULT	NEW::Basic app to do DID, or supply dialtone.
fax_hop_on	Script to talk to a fax redialer
clid_authen	Authenticate with (ani, dnis)
clid_authen_collect	Authenticate with (ani, dnis), collect if that fails
clid_authen_npw	Authenticate with (ani, NULL)
clid_authen_col_npw	Authenticate with (ani, NULL), collect if that fails
clid_col_npw_3	Authenticate with (ani, NULL), and 3 tries collecting
clid_col_npw_npw	Authenticate with (ani, NULL) and 3 tries without pw
SESSION	Default system session application
hotwo	tftp://hostname/scripts/nb/nb_handoffTwoLegs.tcl
hoone	tftp://hostname/scripts/nb/nb_dohandoff.tcl
hodest	tftp://hostname/scripts/nb/nb_handoff.tcl
clid	tftp://hostname/scripts/tcl_ivr/clid_authen_collect.tcl
db102	tftp://hostname/scripts/1.02/debitcard.tcl
*hw	tftp://171.69.184.xxx/tr_hello.tcl
*hw1	tftp://san*tr_db

```
tftp://171.69.184.235/tr_debitcard.answer.tcl

TCL Script Version 2.0 supported.
TCL Script Version 1.1 supported.
```

Note In the output shown, an asterisk (*) in an application indicates that this application was not loaded successfully. Use the **show call application voice** command with the *name* argument to view information for a particular application.

Step 2 Enter the **show dial-peer voice** command with the *peer tag* argument and verify that the application that is associated with the dial peer is correct.

Step 3 Enter the **show running-config** command to display the entire configuration.

TCL IVR Configuration Examples

Use the **show running-config** command to display the entire CUBE configuration Figure 1: Example Configuration Topology, on page 12 shows the type of topology that is used in the configuration for the example.

In this example configuration, CUBE is running TCL IVR for phone A.

This section provides the following configuration examples:

Figure 1: Example Configuration Topology

Phone A   PBX   Cube