IVR setup for Mizu VoIP servers

Contents

About

The Mizu VoIP server has its own embedded flexible IVR functionality which you can use to perform numerous tasks:

- calling card business
- callback access
- auto attendant
- incoming callcenter/contact center
- sales access
- information inquiries
- incoming product support
- and many others

The IVR usually works in the following way:

- acquire a DID number which is routed to your server
- setup an access number for that DID (create a simple Enduser with username set to DID) and assign a campaign to this user
- assign a script to this campaign and create your own script or user/modify one of the built-in templates (from the "IVR" form)
- users will dial this access number and your IVR script will start usually playing various voice files (welcome, your credit is x, etc), reacting to user DTMF input (jump to script entries based on user input) and performing various actions such as call forward, pin recharge and others)

Note: For simple voice prompt before calls you should avoid to use the IVR module because it's higher resource utilization. For this function you can use the server setup wizard to select one of the available prompts such as announcing the user credit and similar prompts. The mizu IVR is sometime referred as "Inbound Callcenter". (The opposite of the "Outbound callcenter" which means the process of placing a high amount of outbound calls (e.g. telemarketing) usually by operators (agents) under Supervisor administration using a special CRM and usually the Predictive Dialer. This functionality is implemented in separate modules and not discussed in this document)

Building IVR Menu's

To build an IVR, you have to follow these steps:

1. Voice files

The mizu voip server is shipped with default voice files that can be used for the IVR. You might replace these sound file with yours or add new files for your IVR needs:
Upload the required voice files to your server in the "voices" directory (add or overwrite the existing ones). The sound files should be standard 8 kHz 16 bit mono PCM files (128 kbits - 15 kb/sec) and are stored in the "voices" subfolder inside the app directory. You can copy-paste the files manually into this directory or you can upload the files with ftp and then a "syncmedia" command must be sent from the server console. Alternatively you can use the "Media Files" form in the MManage to create and upload the voice files.
Test files can be quickly created [here](#).
Note: you will find other recorded files in your voices directory stored in zip packages that might better suite your needs. (replace the voice files with the zipped content if you wish to test them)
Note starting with version 8.8 you can upload wav or mp3 file in any format (the server will automatically convert it to the desired format if needed)

2. Build the IVR

For this you have to use the IVR form.
To create a new IVR script click on the "Add" button. You will find a detailed description about the items below in this document. Alternatively you can edit an existing one or import it from another server.
By default you will have a few scripts that might fit your needs. Make sure you fix the files for all "play file(s)" events.

If you have modified an existing IVR script than the "reload" command must be sent from the server console for the changes to take effect.

3. Create a campaign

Create a new campaign entry using the Campaigns form. You have to assign a unique name and then assign the newly created IVR script menu to the campaign (Campaigns form -> Details tab -> select Script). The rest of the settings can be left with the default values

4. Create the access number

You need to add an Enduser using the "Users and devices" form. Then set its "ivrid" field to the ID of the campaign. This can be done from the "Users and devices" form -> "Functions" tab -> "Campaign ID" field. This will transform the end-user entry to an "access number". If the access number will be used for callback handling, then you need to select the appropriate "Callback access" entry. Make sure that your DID number provider will send DTMF using the INFO or RFC2833 methods (and not In-Band!)

You can list all your IVR access number by selecting the proper filter from the drop-down list on the top of the "Users and devices" form.

IVR Script

Most of the work involved by creating an IVR is to build the IVR script. This can be done using the "IVR" form below "Callcenter" section. On the left side of the form you can find the ivr menu items and you can associate an "action" with each item on the right side. Each action it the right side has a "**jump to**" field, where the IVR will go when it was successfully completed the current action. Most of the action has a "**on fail**" or "**on timeout**" action where you can redirect the script on a failure (like playing a corresponding prompt).

Other action parameters are the following:
Length (min-max): used for dtmf input actions to define the required input length
Timeout (digit-global): used for dtmf input actions to define the key press timeouts
Eof: end character for dtmf input. Usually #.
The item to jump after DTMF can be set by using the list box items. The first item should be * which will define the "default" action.
SQL: used for sql conditional actions to define the SQL text. The IVR will continue on the "jump to" item on successful execution and if the returned value is greater than 0 (first record, first row), otherwise it will jump to the action set on the "on fail" field.
Repeat: used for file playback to set the looping
Wait for playback to finish checkbox: used for file playback if to jump immediately to the next item when the playback have started or wait to complete before to jump. Before DTMF input action usually this is unchecked.
Stop pending voice player: can be used for any action to stop any pending file playback (file play without the "wait for playback to finish" checked). On new file playback or when RTP is received all file playback is stopped automatically without the need to check this field.
End code: can be used to track the ivr items traversed by the users.
In most if the fields (input, sql, etc) the special keywords are allowed which will be replaced automatically to the corresponding values (see below)

IVR Actions

The following actions can be defined at server side:
Wait for DTMF: wait until match or eof characters found or timeout (if no match than it will go to the * entry if any, or to the first entry).
Clear Dtmf: clear any previously received dtmf digits. This should be used before any new dtmf is requested especially if the "ivrautocleardtmfdigits" global config option is set to 0
Validate Input: used to filter out invalid input (pin or destination number, etc)
Store DTMF: the currently entered dtmf will be stored and can be used in a later action using the dtmfstored keyword
Text to speech: speak any text using the built-in TTS engine

Play a File: play any voice message. A repeat count can be defined. You can choose to jump to next action when the play begins, or only when finished (the client will have to listen the whole message before he can move forward with dtmf)
If you uncheck the "Wait for playback to finish", than the IVR will continue with the next action immediately and the playback will be stopped only when a "stop pending playback" condition is reached or a new playback is started.
Play Files: the server can concatenate multiple files separated by a comma
Play Credit: play the credit after the configuration (language, digits, etc)
Play Digits: play a number digit by digit
Play SQL Result ByDigit: will play the returned result digit by digit
Play Date: announce date – time
PlayDTMF: play dtmf digits
StopDTMFIn: don't accept more dtmf input from the user
Play Number: any positive number can be converted to speech until 999999999 and including 0. The basic number files must be created and put in the voices directory. The following files must be present:
zero,one,two,three,four,five,six,seven,eight,nine,ten,eleven,twelve,thirteen,fourteen,fifteen,sixteen,seventeen,eighteen,nineteen,twenty,thirty,forty,fifty,sixty,seventy,eighty,ninety,hundred,thousand,million,and,one
Play SQL Result: you can write any sql query here. The first field in the first column will be played. The SQL can contain keywords, defined below. For example to play the credit for the current user you have to write an SQL like this: select credit from tb_users where id = [calleruserid]
Forward to Phone Number: forward the caller to any number (can include keyword)
Forward to Group: forward the caller to a group (to the best user in that group)
Forward to Group: forward the caller to an operator that belongs in the selected campaign
Forward to SQL: forward the caller to the phone number or username returned by custom sql
        Forwarding will be tried "retry" count in every "forwardretrytimer" sec.
IVR Callback: a callback will be initiated to the previously entered dtmf number
CallingCardAuthentication: perform a PIN card authentication based on the previously entered DTMF digits
AnumAuthentication: perform A number based authentication (ANI). The A number can be rewritten here based on the "ivrfwduser" global config option.
Charge: unused (use "execute sql" for charging)
Mailbox: unused (use "execute sql" and the "record" actions to implement a custom mailbox)
HTTPRequest: make an external http (soap, json, etc) request and store the result (note: the result can be parsed with store sql and conditional sql actions)
SendEmail: will send an email
SendSMS: will send an SMS
Conditional SQL: with this action you can dynamically influence the next ivr menu to be executed. You can enter any SQL query here also using the keywords listed below. The first column first row will be checked. If this column is a value higher than 0 then the next action will be the "jump to" item. Otherwise the next action will be the "on fail" item.
Execute SQL: execute any sql command (check allowed keywords below)
Store SQL Result: will store the sql query result (can be used in further action with the keyword)
Check counter: can be used to detect too many loops (for example to many authentication request)
CheckMaxLoop: same as previous
ResetMaxLoops: reset max loop
FinishAtConnect: use this action (before any call forward action) if the IVR should finish its execution when the B-leg call will be connected
AfterCallConnect: specify IVR item to jump to when the b-leg call is connected (should be used before any call forward action)
AfterCallDisc: specify IVR item to jump to when the b-leg call is disconnected (should be used before any call forward action)
AfterCallFail: specify IVR item to jump to when the b-leg call is failed (should be used before any call forward action)
Start ClientSide Call: use this action to initate the b-leg call. This must be used if the "ivrimmediateclientsidecall" global config option is set to 0. This means that after the routing the IVR can do some other tasks before the actual call forward (for example to play the maximum call duration for the user)
Disc ClientSide Call: will disconnect the b-leg call (for example on a specific dtmf digit)
Record Voicemail: will record the received rtp to the voicemail
Skip Voicemail: skip voicemail record

Next Voicemail: play next voicemail record
Prev Voicemail: play previous voicemail record
Delete Voicemail: delete current voicemail record
Delete all Voicemail: delete all voicemail records
Send Voicemail by email: will send the current voicemail immediately by SMTP
Will email voicemail: will send the recorded voice after call disconnect
Record File: record the conversation to the specified file
Save File: close and save the current recording. Keywords can be used to specify the filename. For example:
rec_[calleruserid]_[currdatetime]
Delete File: terminate recording and drop the current recording file
Start Recording: will (re)start recording the current conversation
TCPRequest: tcp request to the specified IP:port and store the answer to [storedivrstring]
HTTPRequest: http request to the specified server and store the answer to [storedivrstring]
        For the request a .txt file is also accepted.
        The first line might be set to "GET", "REQ", "POST" or "SOAPACTION: action"
Send Email: will send an email via SMTP to the specified address
Send SMS: will send SMS message to the specified phone number
Store Clear: clear the storedivrstring variable
Store Load: load the storedivrstring from file
Store Save: save the storedivrstring to file
CustomX: custom action (specific development for your project if any)
Run: run any batch or executable with the specified parameters
Transfer Credit To: used to transfer credit between endusers
Transfer Credit From: used to transfer credit between endusers
ANI Add: to allow users to add/delete pin less numbers
ANI Remove: to allow users to add/delete pin less numbers
ANI Next: to allow users to add/delete pin less numbers
ANI Store: to allow users to add/delete pin less numbers
WaitToComplete: wait for the current operation to finish
SetLanguage: change language (special handling for the number to speech converter)
Disconnect: will disconnect the call session
Finished: finish with the ivr. No ivr action will be executed after this point

Fields

-**Questions**:
  represents the ivr menu's
-**Order**
  when we specify Next or Prev as "Jump To" actions, than the next or previous record will be determined based on this order.
  also it simplifies the management of the ivr (ivr items can be arranged in a logical order)
-**Alias**
  the name of the ivr item (lowercase and unique). Instead of using ID numbers this text can be easily remembered
-**If**
  conditional sql statement. Ivr items can be skipped based of the result.
-**Go To .. Else**
  Action to do upon the "If" statement
-**entercondition**:
  simple sql clause. If the condition result is true, then it will jump to question 'onconditiontrue', otherwise it will jump to 'onconditionfalse'
-**Repeat**:
  how many times repeat the playback of the specified files
  how many times the call forward will be tried
-**Timeout**:

specify dtmf wait timeout

The first entry is for inter-digit timeout. The second entry is the global timeout.

-**Wait for playback to finish**:

used for voice playback. The ivr playback can start the playback and immediately jump to the next menu, or start the playback and wait for it to finish and only after finished will jump to the next specified menu

-**Stop pending voice players**:

If checked than any current player will be stopped. Otherwise any started file playback will continue to run (until the next "wait for playback to finish" or until timeout or until the "repeat" counter is reached)

-**End code**:

The endcode for the last answer (with a valid code) will be stored for statistical reasons. If you specify more endcodes, the last endcode will be stored in the database.

This code can be used to build statistics. For example we can count in this way how many users have been reached a specified ivr action

-**Completion question**

If the caller will reach to this menu, we say that the questionnaire is completed (for example a product have been sold in case of sales).

-**Jump to:**

Specify the next ivr menu (first, next, previous, or specify exactly).

-**On Timeout:**

Specify the ivr menu to jump to on timeout.

-**Length:**

Minimum and maximum length for dtmf input. When the maximum digits have been entered, the ivr will jump to the next item specified as "Jump To"

-**Eof:**

End of dtmf mark.

The ivr will jump to the next item specified as "Jump To" when this character have been entered.

-**DTMF input text:**

It is a list of digits and assigned actions.

If no match was found, the ivr will jump to the "*" item or to first item if "*" was not found.

-**Forward with Transfer:**

-forwarding can be made completely transparent for the caller (no signaling and media will change and the media will be routed as before)

if this checkbox is checked, the forwarding is done with sip transfer (REFER method)

-**Maxcount**

You can limit how much time an action can be executed (to prevent endless loops)

Keywords

Keyword can be used in sql statements, file names, http requests and in "forward to" numbers.

Any database field can be used as keyword (specified as tablename.fieldname).

Environment variables

In addition, the following keywords are defined as "environment variables":

- ➢ currentnumber
- ➢ callduration
- ➢ ringduration
- ➢ opusername
- ➢ opname
- ➢ currdatetime
- ➢ currdatetimesql
- ➢ currdate
- ➢ currweekday
- ➢ curryear
- ➢ currmonth

- currday
- currhour
- currmin
- currsec
- calleruserid
- calleduserid
- proxyid
- origcallernumber
- callernumber
- callername
- auth_username
- origcallednumber
- callednumber
- calledname
- companyname
- otherpartyname
- otherpartydisplayname
- otherpartyfullname
- techprefix
- called_norm
- auth_username
- auth_password
- transportip
- fromip
- rtprecip
- transportport
- fromport
- rtprecport
- callertype
- callstate
- cc_clientid
- cc_ccid
- dtmf
- dtmftrimmed
- dtmfstored
- dtmfstoredtrimmed
- maxspeachlen
- creditdurationmin
- creditdurationminmin
- creditdurationminminpx
- rating
- ratingcurrency
- rating_tospeech_ex
- dtmfstored
- dtmfstoredtrimmed
- storedivrstring
- storedani

Table name abbreviations:

camp => tb_cccampaigns
client => tb_cclient
campaign => tb_ccampain_clients

script => tb_ccscripts with answertext
scriptcode => tb_ccscripts with code
scriptquestions => tb_ccscripts
scriptanswers => tb_ccscript_answers
quota(s) => tb_ccquotas

"Entercondition" rules:

1. any sql in the following format: select … (if return 0 than the result will be interpreted as false, otherwise as true)
2. simplified condition. can use the following keywords:
prefixes:
        any database table name (tb_cclient, tb_ccampain_clients, etc)
script (maps to scriptquestion.alias - scriptanswer.alias)
client (maps to tb_cclient)
cclient (maps to tb_ccampain_clients)
campaign (maps to tb_cccampaigns)
quotacount, quotapercent, completedcount, completedpercent, averagequotacompletition
suffixes: script alias names or table fields name
operators and functions: most of sql92 keywords will work (=, <>, <, (,),TRIM, LOWER, AND, OR, etc)
examples:
1. (script.alias = 'clinton' or scriptcode.alias = 3 or quotacount.quotaalias >0) and (LOWER(client.name) = 'john')
2. quotacompletedpercent.quotaalias < 100
see the keywords section for more details!
onconditiontrue: if the entercondition result is true, than the next question will be the question with this order (ordernum)
onconditionfalse: if the entercondition result is false, than the next question will be the question with this order (ordernum)

DTMF

H323 gateways support the following DTMF send/receive methods:
-as Q931
-as String
-as Tone (In-Band)
-as RFC2833
SIP endpoints support the following DTMF send/receive methods
-INFO method
-as Tone (In-Band)
-as RFC2833
DTMF in GSM network (send/receive):
        -InBand DMTF
The SIP Softswitch (server) will parse DTMF received:
-INFO method
-as RFC2833
-convert from the above modes to "in-band" dtmf if needed

Usually In-Band DTMF are supported by any vendors in endpoint devices but will not be parsed on IVR servers because of high computation requirements to decode the encoded RTP channels. Make sure that your DID number provider will send DTMF using the INFO or RFC2833 methods (and never In-Band audio)

Authentication

For IVR calls the server will do a "callingcardauth" global config option based authentication.

Please note that in this case the caller device is already authenticated based on basic authorization settings. The IVR needs to find an enduser to allow further operation, like call forward.

The server can authenticate the user based on the following methods:

➢ ANI/CLI authentication: if the CLI is known and this method is allowed.

A number authentication can be used to try user authentication for a call coming from a traffic sender. If user is found with the actual A number then the caller will be authenticated as the enduser, otherwise will be authenticated as traffic sender. In this case you can require a PIN number from the user

Configuration options:

anumberhandling global configuration

- 0=disabled
- 1=only add
- 2= only accept
- 3=add and accept (default)
- 4=only a number access (no pincode request)

enableanumberlookup per user configuration. You need to set it to 1 for traffic senders.

A numbers can be also registered by the users on a web interface or by sending an SMS in the proper predefined format.

➢ PIN (calling-card) based authentification:

When the A number is not known or the A number based authentication is disabled, the IVR have to ask the user for a valid PIN code.

This can be done by the CallingCardAuthentication IVR action.

After the server collects the DTMF digits it will lookup the database for a valid user entry. The authentication can be based on username, password or username+password or depending on the "**callingcardauth**" global config option which can have the following values:

- 0= all combinations with min length checks (default)
- 1=callingcard username
- 2= callingcard password
- 3= callingcard username+password
- 4=any username
- 5=any password
- 6=any username+password
- 7=any username+password
- 8=username for callingcard and username+password or username+pin for other users
- 9=pin
- 10=password or pin
- 11=all combinations
- 12= all combinations with min length checks (default)

The password combined with username or pin works only if no securepasswors are used. It is not secure to pass the password via IVR (better to use PIN)

When using the pin field based authentication, make sure that user has valid pin codes set in this field (when the users are automatically generated, the pin is set to be username+password).

Billing

CDR's generated based on "ivrbilling" global and user setting: 0: one CDR including the forwarded call, 1: load duration only from forwarded call, 2: generate 2 CDR records (A leg + B leg), 3=both merged,4=merged with short a-leg,5=only b-leg billing if call is connected

- ➢ ivrbilling is 0: (server side) 1 CDR will be generated with total client call duration. The billing will be done after the final called user (the IVR accessnumber when the call was not forwarded. Otherwise the final destination number)
- ➢ ivrbilling is 1: (client side) 1 cdr will be generated. The call duration will be set after B-leg call duration and billed accordingly
- ➢ ivrbilling is 2: (both) 2 (or more) cdr will be generated (when there was call forwarding action). The 2 cdr record can be billed separately after different billing tables
- ➢ ivrbilling is 3: (both merged) 1 cdr will be generated, but the enduserprice can be loaded from different billing tables (2-leg merged)
- ➢ ivrbilling is 4: (both merged with short a-leg) 1 cdr will be generated, but the enduserprice can be loaded from different billing tables (2-leg merged). The A leg duration is shortened (only the time spent with IVR until the call forward action)
- ➢ ivrbilling is 5: (server side if connected –mostly the same like ivrbilling 0) 1 CDR will be generated. If the call was not connected then all duration will be billed (you can setup different billing for these calls by marking the entry as "is ivr call" and set the "called" to the access number. If the call is connected, then the B-leg will be billed (possibly after a different billing packet)

Running external applications

Set the datainputtype to "Run".
Specify application name and parameters in the "Command" field.
Keywords can be used as part of command.
The following Parameters are defined (separate them with comma):
  -hide: will launch the application hided
  -wait: will wait for the application to terminate.

Handling Incoming calls

Incoming calls can be handled in different ways:
- ➢ Simple distribution across operators
- ➢ By specifying a callbacknumber and setting the callbackhandling option.
- ➢ Handling by the IVR
- ➢ Make incoming campaigns

Incoming callers (clients) identity can be loaded from database when the call arrives. If the client data is not found, then the client data can be added to the database automatically.

The following config settings will be applied:

CampType:
  0 or NULL=default
1=callcenter (OUT)
  2=IVR (IN) -new numbers will not be requested from the server
  3=Mixed -incoming calls can be received from ivr, but from dialer too

Callbackhandling:
0=dropp all
1=route to callbackroutenumber
2=route to free operators
3=route to free operators, and if not answered than route to callbackroutenumber

4=first to callbackroutenumber than to free operators
Callbacknumber:
    "A" number for calls. For example the predective dialer will use this number
Callbackringtimeout:
    ring timeout on callback (after than play ivr message if set)
defrecallmin ring timeout on callback (after than play ivr message if set) defrecallmin
callbackroutenumber:
    number to be dialed on incoming calls when callbackhandling is 1,3 or 4
ivr_addmissingusers: -applicable on serverside ivr calls and in MAgent
0=no -default
1=add to client table
2=add to campaign client
addmissingusersto: -in wich campaign to add the incoming client
    0=add to the current campaign (default)
    other=add to the specified campaign (campaign id)
Handlingincoming:
  0=not handled
  1=allow to search the current campaingn for the incomig number
  2=allow to search and edit the current campaingn for the incomig number
  3=allow to search the whole database for the incomig number
  4=allow to search and edit the whole database for the incomig number
  5=show scripts form

Incoming campaigns:
1. First you must define your phone numbers which will handle the incoming campaign(s).
2. If the server is a virtual server, you must setup your main server to route that numbers to the virtserver (See this caption for more details)
3. You must add this numbers to your user list, and assign an IVR for them.
4. For the IVR campaign set the "CampType" field to 2 and don't forget the ivr_addmissingusers and handlingincoming settings
5. Create the MAgent script and GUI in the MManage.
6. Assign some operators for the campaign(s).

Call-Back services

Callback services can be implemented by entering the number to call on the website or by CID or ANI callback.
To define access numbers set tb_user.iscallback to the required ivr (campaignid) and tb_user.anumberlookup to 1.
The calls will be authenticated based on "ivrauthentication" global config value (0=no,1=A number or pincode, 2=only A number, 3=only pin code, 4=A number and pincode)
When using only A number authentication be aware of fraud possibilities.

Callback will be initiated from "callbackcallernumber" A number or from the user who received the call if "callbackcallernumber" is not specified.
The callback will automatically start the IVR specified in the "iscallback" field or in "ivrid" field of the access number.

Calling Card services

For a calling card service you must setup an IVR (and a campaign) that will handle the authentication and forward the call to the requested destination.
To create an access number, add a "power user" and set the ivrid to the campaign id you wish to use.

You can also allow authentication based on A number by setting "anumberlookup" for the traffic sender.
Example IVR:
  -Play File: Welcome to xxx. Please enter your PIN number
  -Wait for dtmf (acquire PIN digits)

-CallingCardAuthentication: (Will check the pincode in the database). On success go to next item, on fail play a prompt to try again.
-Play File: Please enter the destination number
-Forward to phone number [dtmftrimmed] (call forwarding)
-Finish (release ivr)

If you want to bill the user for calling the ivr, then set the "resetdurationonfwd" global config to "false". Otherwise set to "true" and only the forwarded call will be billed.

PIN codec can be stored in the user table. Their type must be set to 0, isoperator to 6 and authentication based on username (5)

Pin codes

Recharge codes used if you have prepaid cards printed.
You can generate random prepaid codes from MManage -> Billing -> Pincodes form

Prepaid account can be charged over the website or by ivr:
IVR operation:
-automatic user authentication based on sip registration or require entering the phone number to be charged
-require pincode
-check if pincard is valid (tb_prepaidcodes)
-increase credit for the user (tb_users.credit)
-goodbye message

The same prepaid codes can be used also from the web user interface.

Load and store

You can store the results of various operations in the **storedivrstring** variable and then use it later in other.
The results are stored for the following actions:
- o Store SQL Result
- o HTTPRequest
- o TCPRequest
- o Run
- o Store Load

Then you can use the [storedivrstring] anywhere later to use the stored value.
For example use it in the followings:
- o Execute SQL
- o Run
- o HTTPRequest
- o Store Save
- o Play file(s)

You can clear the storedivrstring with the Store Clear action.

Please note that you can easily transform any string by using the replacer.exe command. Example workflow:
1. Run any action which will load something in the storedivrstring variable (a result of an SQL query, a command line application or just load from file)
2. Save the storedivrstring to file (Store Save)
3. Run the replacer.exe or any other tool with any parameter (replace strings in the file)
4. Store Load (load the result)
5. Use the transformed storedivrstring anywhere you wish

Phone to Phone (P2P) calls

You can implement this service on your website or in softphone. The website or softphone should connect to the server console (a TCP connection) and issue the p2p call command. All communication can be encrypted and the clients are authenticated properly.

The billed user will be the logged in user. The p2p command has 3 parameters. A number, B number and IVR id (optional).
2 CDR records are generated for these calls, both of them billed to the initiator (based on your usual pricing).

Language and localization

The default system language is English.
Multiple languages can be implemented using Conditional SQL and SetLanguage actions.
Make sure to have the voice files in the desired language appending the language suffix for the original file names.
For example six_hu.wav should play 6 in Hungarian ("hat").
If you don't wish to use English at all, then you can just overwrite the existing files if you wish.
Instead of voice files, you can also use Text to Speech if your language is supported.
You can switch the language at any time in the IVR by calling the **SetLanguage** action passing the 2 digit language code.

An important part of the translation is the number announcements (usually to play the user credit or remaining time).
The system is capable to playback numbers between 0 and 999999999.
By default it uses the English grammar rules to compose the numbers, however extra customizations can be applied here and you can change the behavior with the following settings:
- numberspeechmethod: the algorithm used to play numbers (-1=default, 1=old, 2=rus,3=rus without currency,4=ro, 20=new (default), 30=TTS)
- language: global config option or can be changed with the SetLanguage action (default is english)
- playcreditprecision: number of digits (default is 2)
- currency: system global or per user setting (for example "USD")
- currencyname: system global currency name (for example "dollar")
- centsname: name of the fractional part (such as "cent" or "penny")

Make sure to have voice files for all the basic numbers: 1-10 (one-teen) or 1-20 (one- twenty), 100 (hundred), 1000 (thousand), 10000 (million) with the language prefix appended. For example hundred_hu should play "száz" in hungarian.
You can also store the files with number file names. The system will first check for worlds (two.wav, four.wav) and if this not found then it will check for 2.wav, 4.wav, etc.

TTS (Text to Speech)

You can use the built-in TTS capabilities to play any text, without the need to have a recorded voice file for it. This might have a bit lower quality, but it can be very useful for arbitrary messages. Use the "[Server] Text To Speech" IVR action with any text you wish.

There are two different engines which can be used for TTS:
- Microsoft SAPI
- eSpeak

You can set the engine to be used with the following tts config option:
tts
It can have the following values:
- -3: TTS disabled
- -2: SAPI
- -1: eSpeak
- empty: auto
- 0-1000: SAPI index (Use the "ttssapi" server console command to find out SAPI index values supported by your system)
- 2 char language code: eSpeak language (for example hu means Hungarian)

You can also set it **per language** by storing a configuration value to the key with the language suffix.
For example:

- tts_en=0 //will use SAPI index 0 to play English texts
- tts_fr=5 //will use SAPI index 5 to play French texts
- tts_ro=ro //will use the eSpeak ro language to play Romanian text
- tts_ro=hu //will use the eSpeak hu language to play Hungarian text

You should use the TTS engine which has support for the language you need (or better quality if supported by both SAPI and eSpeak).

eSpeak has support for the following languages with various quality:
Afrikaans, Albanian, Aragonese, Armenian, Bulgarian, Cantonese, Catalan, Croatian, Czech, Danish, Dutch, English, Esperanto, Estonian, Farsi, Finnish, French, Georgian, German, Greek, Hindi, Hungarian, Icelandic, Indonesian, Irish, Italian, Kannada, Kurdish, Latvian, Lithuanian, Lojban, Macedonian, Malaysian, Malayalam, Mandarin, Nepalese, Norwegian, Polish, Portuguese, Punjabi, Romanian, Russian, Serbian, Slovak, Spanish, Swahili, Swedish, Tamil, Turkish, Vietnamese, Welsh.

eSpeak requires XAudio2_8 to be installed on your system as part of the DirectX API (installed by default on Windows 8 and never but not on Server 2012). The espeak installer is also shipped with the VoIP server binaries: espeak.msi.
More details about eSpeak can be found from here and here.

SAPI by default has support only for English and your system locale language if installed.
However you can add more languages as needed.
If you don't have SAPI installed, you can download from here.
To add more languages, you will either have to install language packs to your system or download SAPI languages from here.

Use the "ttssapi" command to find out SAPI index values supported by your system.

Global or campaign level config options

There are some options that will influente the behavior of the IVR scripts.
This can be found by searching for the "ivr" keyword in the global configurations form. Some scripts might need this values to be set differently.
ivrimmediateclientsidecall: 0=no (default) ,1=yes (usually yes. othervise need to use ivrStartClientSideCall) –available also on campaign level
Usually if this is set to 0 then the ivrmovefwdaftertransf should be also changed to 2.
ivrmovefwdaftertransf: 0=no,1=on succ send to routing,2=after succ routing (default),3=on call connect –available also on campaign level
ivrautocleardtmfdigits: 0=no,1=after actions (default), 2=when beginning dtmf capture, 3=always (if 0 then use ivrClearDtmf actions) –available also on campaign level
ivrtimeout: timeout without b-leg call. Default is 15 (minutes)
ivrtimeout2: timeout if there was a b-leg call. Default is 40 (minutes)
maxivrloop: max ivr action in one session (default is 500)
banivrauth: 0=no,1=callbacks,2=all (default; check max failed ivr pincode)
maxivrauthfail: maximum number of failed authentication requestst
storecscriptanswers: set to 1 to store the IVR script path that was traversed by the users
ivrauthentication: how to authenticate the users on the IVR. See the Admin guide for the details.
ivrbilling: default is 5. See the Admin guide for the details.
ivrfwduser: how to rewrite the A number on authentication: 0=userid is pincode callernumber is orig caller (default), 1= userid is pincode callernumber is pincode,2=orig caller
ivrplaybackstopondtmf: 0=no,1=terminate playback on ivrWaitForDTMF,2=terminate playback on all,3=terminate playback and skip all playback (default)
playivrfwdringtone: 0=no,1=on ring received if other player is not playing (default),2=on ring received always,3=fake

useivrreinvite: wheter to send back a reinvite to offload the media routing (not supported by many devices)
freeivraccess: 0=no (default),1=callback,2=ivr,3=ivr+callback
ivrdtmfmethod: dtmf method to be generated by the ivr when needed: 0=inband,1=info,2=inband+info,3=RFC2833,4=RFC2833 and info,5=inband, RFC2833 and info,6=inband, RFC2833 or info
maxivrthreads: how much tread to be allowed for the IVR
ivrthreadtreshold: how much tread to be allowed for the IVR (threshold for a new thread)
playcreditprecision: number of floating point digits (default is 2)
numberspeechmethod: -1=default, 1=old, 2=rus,3=rus without currency,4=ro, 20=new (default), 30=TTS

Technical description

On routing if calleduser.ivrid > 0 than set ep ivr_id and calltype to eIvr
        No other ep will be created
        ivr_id is the campaignid
On invite (after routed) if calltype = eIvr () call IvrAction(actIvrStart);
IvrAction(EIvrActionType actiontype)
1.   the actiontype will tell what was happened: actDefault, actIvrStart, actReady, actUserInput, actTimeout, actFail
2.   load the scripts if already not loaded
3.   Check next action todo (jumpto). Especially check for dtmf input action
a.   if actiontype == actUserInput than check if input finished (eof or maxlen)
b.   if play finished and need to repeat, than play again
c.   otherwise set jumpto
4.   JumpTo next action if needed (check enterconditions)
5.   execute the next action (datainputtype, if any)
    Play file, execute sql, forward, etc

Notes

-For skill based callcenters you can use the "Forward to Group" or the "Forward to SQL" actions.
-You can assign an IVR menu to any user if you set the "ivrid" field to a valid campaignid id.
-IVRs are directly mapped to campaigns and can be edited in the script editor. Set your CampType to 2 (IVR IN) if you have assigned to an IVR.
-If you need something to do between the call forward and the server side call, then you should set the ivrmovefwdaftertransf to 2 and the ivrimmediateclientsidecall to 0 and then use the "Initiate client side call" IVR action to actually send the call.
-Any kind of voicemail system can be implemented with the voicemail and recording actions. You might prefer to modify one of the existing templates than build from scratch
-List answers will be separated by semicolon ;
-All script answers (including data input) can be saved to tb_ccscript_processing and used for further statistics

For more details, please consult the [Admin Guide](#) and the [IVR wiki](#).
For more help, contact serversupport@mizu-voip.com.

Copyright © MizuTech SRL