PBX Integration

VOIspeed enables you to integrate the PBX service with an external application via the server MySQL. The main elements of the integration are:

- the PBX server
- the MySQL server
- the user application

The PBX server writes all the events surrounding the user, calls, etc in the monitoring table. Then the MySQL server runs, via the http protocol, the URL of the user application with all the Calling numbers necessary to identify the event.

Configuration of the PBX server

Within the settings of the PBX you need to enter the user application URL (reachable via http protocol) that you can find in the Company's properties.

Configuration of the MySQL server

To run the user application you require a client http, in our case cURL, that can be downloaded from http://curl.haxx.se/download.html based on the architecture of the MySQL server (to learn about MySQL architecture follow the query SELECT@@version_compile_os;)

After having installed cURL you need to configure the MySQL server to enable the running of the external programme via plug-in. The plug-in required is supplied with documentation downloaded from the VOIspeed website (in the file lib-mysqludf.sys choose the correct file based on the architecture of your WEB Server) which should be copied into the file of MySQL (usually it is a lib/plugin, but to find the exact location of the directory you can use the query SELECT@@plugin_dir;).

After having copied the file follow the query:

- Windows:
CREATE FUNCTION sys_exec RETURNS int SONAME 'lib_mysqludf_sys.dll';

- Linux:
CREATE FUNCTION sys_exec RETURNS int SONAME 'lib_mysqludf_sys.so';

To finish the MySQL configuration, follow the query supplied in the file mysql_trigger.sql, which creates the trigger and the functions required to run the external application for the user.

Describing SOAP

To run the commands you need to use the login credentials of a user whose credentials allow integration.

The password must be sent as a hash MD5, which can be extracted from the database (table of users):

- auth-First name@domain (username)
- MD5-HASH-PASSWORD (password)

The SOAP header is the same for each command.

```
<?xml version="1.0" encoding="UTF-8" ?>
<soap:Envelope xmlns:soap="http://www.w3.org/2001/12/soap-envelope"
soap:encodingStyle="http://www.w3.org/2001/12/soap-encoding">
<soap:Header>
<First name>auth-First name@domain</First name>
<password>MD5-HASH-PASSWORD</password>
</soap:Header>
```

The SOAP body changes based on the same command, as written below. The part to change is that enclosed between <vspbx:…> e </vspbx:…>

<soap:Body>
<vspbx:Command>
…
</vspbx:Command>
</soap:Body>
</soap:Envelope>

The identifiers (user call ID, device ID, ecc) are shown with a string of 8 characters, for example: 000012A0

Call Request

Request to execute a new call
<vspbx:CallReq>
<param1>*calling-First name@domain*</param1>
<call_req_number>*number*</call_req_number>
<call_req_device_id>*deviceID*</call_req_device_id>
<call_req_ext_id>*externalID*</call_req_ext_id>
</vspbx:CallReq>

Calling number	Description
<vspbx:CallReq>	command
<param1>	User controlling the commands
<call_req_number>	number to call
<call_req_device_id>	ID of the device
<call_req_ext_id>	External ID (STRING) generated by the user (optional)

Call Request Extended

Request to make a new call between different elements of the system

<vspbx:CallReqExt>
<param1>*calling-First name@domain*</param1>
<call_request_calling_id>*calling ID*</call_request_calling_id>
<call_request_called_id>*called ID*</call_request_called_id>
<call_request_calling_type>*calling type*</call_request_calling_type>
<call_request_called_type>*called type*</call_request_called_type>
<call_request_calling_number>*calling number*</call_request_calling_number>
<call_request_called_number>*called number*</call_request_called_number>
<call_request_external_id>*external ID*</call_request_external_id>
<call_request_device_id>*device ID*</call_request_device_id>
<call_request_to_be_rec>*to be recorded*</call_request_to_be_rec>
</vspbx:CallReqExt>

Calling number	*Description*
<vspbx:CallReqExt>	command
<param1>	the user for whom you wish to set the command

<call_request_calling_id>	ID caller
<call_request_called_id>	ID entity called
<call_request_calling_type>	0 = callable_user 1 = callable_group 2 = callable_gateway 3 = callable_ivr 4 = callable_service 5 = callable_lcr 6 = callable_messages 7 = callable_device 8 = callable_wan_user 9 = callable_voicemail 10 = callable_voicemail_manager 11 = callable_meeting_room
<call_request_called_type>	See <call_request_calling_type>
<call_request_calling_number>	Calling number
<call_request_called_number>	Number called
<call_request_external_id>	External ID (STRING) generated by the user (optional)
<call_request_device_id>	ID of the device to use to make a call
<call_request_to_be_rec>	0 = no call recording 1 = call recording

Call Accept

Accepting a call:

<vspbx:CallAccept>
<param1>*calling-First name@domain*</param1>
<param2>*user call ID*</param2>
</vspbx:CallAccept>

Parameter	*Description*
<vspbx:CallAccept>	command
<param1>	the user for whom you wish to set the command
<param2>	ID of the call to accept

Call Hold

Putting a call on hold:

<vspbx:CallHold>
<param1>*calling-First name@domain*</param1>
<param2>*user call ID*</param2>
</vspbx:CallHold>

Parameter	*Description*
<vspbx:CallHold>	command
<param1>	the user for whom you wish to set the command
<param2>	ID of the call to put on hold

Call Retrieve

Retrieving a call on hold:

<vspbx:CallRetrieve>
<param1>*calling-First name@domain*</param1>
<param2>*user call ID*</param2>
</vspbx:CallRetrieve>

Parameter	*Description*
<vspbx:CallRetrieve>	command
<param1>	the user for whom you wish to set the command
<param2>	the ID of the call to retrieve from on hold

Call Divert

Deviazione di una chiamata

<vspbx:CallDivert>
<param1>*calling-First name@domain* </param1>
<call_divert_callid>*user call ID*</call_divert_callid>
<call_divert_recipient>*extension*</call_ divert_recipient>
</vspbx:CallDivert>

Parameter	*Description*
<vspbx:CallDivert>	command
<param1>	the user for whom you wish to set the command
<call_divert_callid>	ID of the call to divert
<call_divert_recipient>	Extension to divert the call

Call Transfer

Transfering a call:

<vspbx:CallTransfer>
<param1>*calling-First name@domain*</param1>
<call_transfer_callidfrom>*user call ID*</call_transfer_callidfrom>
<call_transfer_callidto>*extension*</call_transfer_callidto>
</vspbx:CallTransfer>

Parameter	*Description*
<vspbx:CallTransfer>	command
<param1>	the user for whom you wish to set the command
<call_transfer_callidfrom>	ID of the call to transfer
<call_transfer_callidto>	ID of the call which receives the transfer

Call Disconnect

Disconnecting a call:

<vspbx:CallDisconnect>
<param1>*calling-First name@domain* </param1>
<param2>*user call ID*</param2>
</vspbx:CallDisconnect>

Parameter	*Description*
<vspbx:CallDisconnect>	command
<param1>	the user for whom you wish to set the command
<param2>	ID of the call to disconnect

Call Device Switch

Switching phone device during a call:

<vspbx:CallDeviceSwitch>
<param1>*calling-First name@domain* </param1>
<param2>*user call ID*</param2>
<call_device_switch_id>*device ID*</call_device_switch_id>
</vspbx:CallDeviceSwitch>

Parameter	*Description*
<vspbx:CallDeviceSwitch>	command
<param1>	the user for whom you wish to set the command
<param2>	ID of the call for which the change of device is required
<call_device_switch_id>	ID of the device to switch to

Call PickUp

Picking up a call:

<vspbx:CallPick>
<param1>*calling-First name@domain* </param1>
<call_pick_number>*number*</call_pick_number>
<call_pick_device_id>*device ID*</call_pick_device_id>
</vspbx:CallPick>

Parameter	*Description*
<vspbx:CallPick>	command
<param1>	the user for whom you wish to set the command
<call_pick_number>	Extension picking up the call
<call_pick_device_id>	ID of the device used

Call Recording Start

To start recording a call:

<vspbx:CallRecStart>
<param1>*calling-First name@domain* </param1>
<param2>*user call ID*</param2>
</vspbx:CallRecStart>

Parameter	*Description*
<vspbx:CallRecStart>	command
<param1>	the user for whom you wish to set the command
<param2>	ID of the call with which to begin recording

Call Recording Stop

To stop recording a call:

<vspbx:CallRecStop>
<param1>*calling-First name@domain* </param1>
<param2>*user call ID*</param2>
</vspbx:CallRecStop>

Parameter	*Description*
<vspbx:CallRecStop>	command
<param1>	the user for whom you wish to set the command
<param2>	ID of the call for which you want to stop the recording

Crud object

Forces the PBX to act upon a system's component.

<vspbx:CRUDevent>
<crud_event_action>*action*</crud_event_action>
<crud_event_entity_type>*entity_type*</crud_event_entity_type>
<crud_event_entity_subtype>*entity_sub_type*</crud_event_entity_subtype>
<crud_event_entity_id>*entity_id*</crud_event_entity_id>
</vspbx:CRUDevent>

Parameter	*Description*
<vspbx:CRUDevent>	command
<crud_event_action>	0 = crud_none 1 = crud_create 2 = crud_read 3 = crud_update 4 = crud_delete
<crud_event_entity_type>	0 = cet_none 1 = cet_company 2 = cet_user 3 = cet_group 4 = cet_gateway 5 = cet_ivr 6 = cet_routing_rule 7 = cet_timetable 8 = cet_department 9 = cet_router 10 = cet_service 11 = cet_settings 12 = cet_voicemail 13 = cet_voicemail_manager 14 = cet_service_message 15 = cet_meeting_room 16 = cet_wan_user
<crud_event_entity_subtype>	Only for the type: cet_gateway 0 = gateway_voip 1 = gateway_wan 2 = gateway_isdn 3 = gateway_gsm 4 = gateway_analogic
<crud_event_entity_id>	ID of the event

Description of the tables

monitor_devices

Column	Type	Description
ID	int(10)	Unique ID of the row
companyid	int(10)	Connected to company.ID
instime	datetime	Time of the event
deviceid	int(10)	ID of the device
userid	int(10)	Connected to users.ID
devicety	tinyint(3)	Type of device: 0 = unknown_device 1 = gui_device 2 = sip_phone 3 = usb_device 4 = dect_device 5 = mobile_device 6 = video_device
host	varchar(80)	URL of the device (leave empty for devicety = 5)
descript	varchar(80)	Description of the device (mobile number for the devicety = 5)

monitor_calls

Column	Type	Description
ID	int(10)	Unique ID of the row
companyid	int(10)	Connected to company.ID
instime	datetime	Time of the event
starttime	datetime	Call start time
callid	int(10)	Call ID
callingid	int(10)	ID of the caller, connected with various tables based on callingty
callingty	tinyint(3)	Type of call: 0 = callable_user (callingID connected with users.ID) 1 = callable_group (callingID connected with group.ID) 2 = callable_gateway (callingID connected with gateway.ID) 3 = callable_ivr (callingID connected with ivr.ID) 4 = callable_service (callingID connected with service.ID) 5 = callable_lcr (callingID dummy generated runtime) 6 = callable_messages (callingID connected with service_messages.ID) 7 = callable_device (callingID connected with monitor_devices.deviceid) 8 = callable_wan_user (callingID connected with remote_wanusers.ID) 9 = callable_voicemail (callingID connected with voicemails.ID) 10 = callable_voicemail_manager (callingid connected with voicemails.ID) 11 = callable_meeting_room (callingID connected with meeting_room.ID)
callingdsc	varchar(80)	Description of the caller
callingnr	varchar(20)	Caller number
calledid	int(10)	Caller ID, connected to different tables based on the calledty
calledty	tinyint(3)	Type of call (see callingty)
calleddsc	varchar(80)	Description of the person called

callednr	varchar(20)	Number of the person called
connected	int(10)	ID of the called person, connected with different tables based on connectty
connectty	tinyint(3)	Type of the call made (see callingty)
connectdsc	varchar(80)	Description of the call made
media	tinyint(3)	Type of call media 0 = m_audio_video 1 = m_chat
recording	tinyint(3)	Indicates if the call is recorded (by selecting gateway option) 0 = NO 1 = YES
status	tinyint(3)	Call status 0 = com_idle 1 = com_incoming 2 = com_calling 3 = com_active 4 = com_diverting 5 = com_picking up 6 = com_rejecting 7 = com_terminating 8 = com_terminated
externalid	varchar(40)	ID in relation to the call (generated by the user)

monitor_gateways

Column	Type	Description
ID	int(10)	Unique ID for the row
companyid	int(10)	Connected with companies ID
instime	datetime	time of the event
gatewayid	int(10)	gateway ID
type	tinyint(3)	Type of gateway 0 = gateway_voip 1 = gateway_wan 2 = gateway_isdn 3 = gateway_gsm 4 = gateway_analogic
name	varchar(50)	gateway description
ipaddr	varchar(15)	gateway IP
port	smallint(5)	gateway port
realm	varchar(255)	realm of the gateway
lines	int(10)	total number of lines
linesbusy	int(10)	lines busy
status	tinyint(1)	connection status 0 = inactive 1 = active
enabled	tinyint(1)	Indicates 0 = disabled 1 = enabled

monitor_gw_users

Column	Type	Description
ID	int(10)	Unique ID for the row
companyid	int(10)	Connected with companies.ID
instime	datetime	Time of the event
wanuserid	int(10)	Connected with remote_wanuser.ID

gatewayid	int(10)	Connected with con gateway.ID
dept	varchar(80)	Department for the user
surname	varchar(40)	Surname of the user
name	varchar(40)	First name of the user
extension	varchar(20)	Internal number for the user
avstatus	tinyint(3)	Audio/ video status of the user 0 = status_unknown 1 = status_unreachable 2 = status_available 3 = status_ringing 4 = status_busy 5 = status_dnd (non disturbare) 6 = status_sleeping 7 = status_absent
chstatus	tinyint(3)	Status of the user GUI (see avstatus)

monitor_users

Column	Type	Description
ID	int(10)	Unique ID for the row
companyid	int(10)	Connected with companies.ID
instime	datetime	Time of event
userid	int(10)	Connected with users.ID
deptid	int(10)	Connected with company_department.ID
surname	varchar(40)	Surname of the user
name	varchar(40)	First name of the user
extension	varchar(20)	Internal user extension
avstatus	tinyint(3)	Audio/ video status of the user 0 = status_unknown 1 = status_unreachable 2 = status_available 3 = status_ringing 4 = status_busy 5 = status_dnd (do not disturb) 6 = status_sleeping 7 = status_absent
chstatus	tinyint(3)	Status of the User GUI (see avstatus)
numcalls	tinyint(3)	Number of active calls with the user
mode	tinyint(3)	Function mode of the user 0 = mode_normal 1 = mode_dnd (do not disturb) 2 = mode_absent

monitor_user_calls

Column	Type	Description
ID	int(10)	Unique ID for the row
companyid	int(10)	Connected with companies.ID
instime	datetime	Time of event
usercallid	int(10)	Call ID (to use with SOAP commands)
callid	int(10)	Connected with monitor_calls.callid
userid	int(10)	Connected with user.ID
deviceid	int(10)	Connected with monitor_devices.device_id
direction	tinyint(1)	Direction of the call 0 = no_direction

		1 = incoming_call 2 = outgoing_call
status	tinyint(3)	Call status: 0 = call_status_none 1 = call_status_connecting 2 = call_status_active 3 = call_status_hold 4 = call_status_voice_mail 5 = call_status_parked 6 = call_status_closed 7 = call_status_trying 8 = call_status_killme 9 = call_status_early_connecting
starttime	datetime	Date/time the call begins
party_nr	varchar(20)	Number of the correspondent
party_desc	varchar(255)	Description of the correspondent
party_gway	varchar(50)	Name of gateway used
group	varchar(20)	Group called
ans_btn_en	tinyint(1)	Status of the button "call answer" 0 = disabled 1 = enabled
gui_cmd_en	tinyint(1)	Status of GUI commands: 0 = reject commands from the GUI 1 = accept commands from the GUI
recording	tinyint(1)	Call recording status: 0 = call recording not yet begun 1 = recording begun
callqueued	tinyint(1)	Call queuing status: 0 = call not in queue 1 = call in queue
dev_switch	tinyint(1)	Status of the request to change device: 0 = no request made 1 = request for change in progress
dev_sw_id	int(10)	ID of device used to end the exchange Connected with monitor_devices.device_id
externalid	varchar(40)	External id of the call (generated by the user)

users

Column	Type	Description
ID	int(10)	Unique ID for the row
companyid	int(10)	ID of the company to which the user belongs
First name	Varchar(80)	First name which identifies the user in the system
password	Varchar(32)	User password for the GUI and user area
passwd_sip	Varchar(32)	User password for the SIP telephones
otp_pwd	Varchar(255)	RESERVED AREA
otp_expire	datetime	RESERVED AREA
act_token	Varchar(255)	RESERVED AREA
act_pwd	tinyint(3)	RESERVED AREA
pincode	Varchar(10)	Pin used by the telephone keypad (max 6 digits)
extension	Varchar(6)	Extension number
name	Varchar(40)	First name
surname	Varchar(40)	Surname
deptid	Int(10)	ID of the department
numoffice	Varchar(40)	Office number
numhome	Varchar(40)	Home number
nummobile	Varchar(40)	Mobile number

numfax	Varchar(40)	Fax number
numother	Varchar(40)	Other number
email	Varchar(80)	Email address
level	Tinyint(3)	Level of user: 1 = super administrator 2 = administrator 5 = normal user
status	Tinyint(3)	Fixed to 1
forbidden	text	List of blocked prefixes, separated by a comma
allowed	text	List of those prefixes always allowed, separated by a comma
caps	Int(10)	User permissions Bit Mask 1 = cap_none 2 = cap_chat 4 = cap_file_xfer 8 = cap_call_recording 16 = cap_dnd 32 = cap_absent 64 = cap_vocalbox 128 = cap_call_xfer 256 = cap_whocalledwho 512 = cap_closures 1024 = cap_divert
exitnum	Varchar(40)	Outgoing number
admincaps	Varchar(16)	Administrator permissions for the web configurator
av_mode	Tinyint(3)	User audio/video mode: 0 = mode_normal 1 = mode_dnd 2 = mode_absent 3 = mode_divert
chat_mode	Tinyint(3)	GUI user mode: 0 = mode_normal 1 = mode_dnd 2 = mode_absent 3 = mode_divert
usemob_cal	Tinyint(3)	Use mobile to receive calls directly (0 = NO, 1 = SI)
usemob_grp	Tinyint(3)	Use mobile to receive calls from group (0 = NO, 1 = SI)
maxringtim	Int(10)	Maximum ring time (seconds)
phrase	Varchar(255)	Status phrase (slogan)
callonbusy	Tinyint(3)	Receive the call even if busy (0 = NO, 1 = SI)
evnone_act	Tinyint(3)	NOT USED
evnone_num	Varchar(20)	NOT USED
evnone_src	Varchar255)	NOT USED
evbusy_act	Tinyint(3)	Action to take ‘on busy’: 0 = call_action_none 1 = call_action_reject 2 = call_action_vocalbox 3 = call_action_playback 4 = call_action_transfer 5 = call_action_queue
evbusy_num	Varchar(20)	Event: “on busy”, number to call
evbusy_src	Varchar(255)	Event: “on busy”, audio file source
evunre_act	Tinyint(3)	Event “unreachable”, azione da compiere (see evbusy_act)
evunre_num	Varchar(20)	Event “unreacheable”, number to call
evunre_src	Varchar255)	Event “unreacheable”, audio file source
evmaxr_act	Tinyint(3)	Event “maximum ring time”, action to be taken (see evbusy_act)
evmaxr_num	Varchar(20)	Event “maximum ring time”, number to call
evmaxr_src	Varchar(255)	Event “maximum ring time”, audio file source
evmaxc_act	Tinyint(3)	NOT USED
evmaxc_num	Varchar(20)	NOT USED

evmaxc_src	Varchar(255)	NOT USED
evmaxq_act	Tinyint(3)	NOT USED
evmaxq_num	Varchar(20)	NOT USED
evmaxq_src	Varchar(255)	NOT USED
sel_route	Varchar(255)	WAN route selected from the GUI
sel_depart	Varchar(40)	Deparatment path selected from the GUI
lost_last	datetime	Date/ time of the last visit from the missed calls list
chat_last	datetime	Date/ time of the last visit on the chat function
vbox_last	datetime	Date/time of the last visit to the voicemail
int_enable	Tinyint(3)	Integrated on the GUI enabled (0 = NO, 1 = SI)
int_sh_win	Tinyint(3)	NOT USED
int_onlyex	Tinyint(3)	Enable Integra only for external numbers (0 = NO, 1 = YES)
int_calin	Tinyint(3)	Enable Integra on the incoming call event (0 = NO, 1 = YES)
int_calfw	Tinyint(3)	Enable Integra on the call forwarding event (0 = NO, 1 = YES)
int_calout	Tinyint(3)	Enable Integra on the outgoing call event (0 = NO, 1 = YES)
int_calacc	Tinyint(3)	Enable Integra on the call accepted event (0 = NO, 1 = YES)
int_calclo	Tinyint(3)	Enable Integra on the call closing event (0 = NO, 1 = YES)
int_calact	Tinyint(3)	Enable Integra on the active call event (0 = NO, 1 = YES)
int_http	Tinyint(3)	NOT USED
int_shell	Tinyint(3)	Execute Integra command via Windows Shell (0 = NO, 1 = YES)
int_cmd	Varchar(255)	String of Integra command executed via Windows Shell
blind_enab	Tinyint(3)	Enable high contrast screen from the GUI (0 = NO, 1 = YES)
cwi_enab	Tinyint(3)	Call Waiting Indication, beep for incoming call (0 = NO, 1 = YES)
custgroups	Text	Personalised groups created from the GUI
url_web	Text	NOT USED
aux_user	Varchar(100)	NOT USED
aux_pass	Varchar(100)	NOT USED
aux_enab	Tinyint(3)	NOT USED
divertnum	Varchar(40)	Number to which to divert, activate if different from the empty string
lastdivnum	Varchar(40)	Number with which to make the diversion (memory)

groups

Column	Type	Description
ID	int(10)	Unique ID for the row
companyid	int(10)	Company ID
name	Varchar(80)	Name of group
descript	Varchar(255)	Description of group
pincode	Varchar(10)	PIN Code for group
vmaddress	Varchar(255)	NOT USED
voicemail	Varchar(80)	NOT USED
ringmode	Tinyint(3)	Ring mode: 0 = ring_priority 1 = ring_all 2 = ring_cyclic 3 = ring_cyclic_shift 4 = ring_cyclic_add 5 = ring_cyclic_shift_add 6 = ring_ACD
skiponbusy	Tinyint(1)	Skip extension if busy (0 = NO, 1 = YES)
maxringtim	Smallint(5)	Maximum ring time
sliptime	Smallint(5)	Slip time between one user and another in the cyclical ring
disabpick	Tinyint(1)	Disable the diversion (0 = NO, 1 = YES)
opwaitenab	Tinyint(1)	Enable operator waiting (0 = NO, 1 = SI)
opwaitdel	Smallint(5)	Operator waiting time
opwaitmsg	Tinyint(3)	Type of message from waiting operator:

		0 = audio_none 1 = audio_file 2 = audio_linein 3 = audio_dynamic
opwaitsrc	Varchar(255)	Message source of operator waiting
busy_act	Tinyint(3)	Event “on busy”, action to be taken: 0 = call_action_none 1 = call_action_reject 2 = call_action_vocalbox 3 = call_action_playback 4 = call_action_transfer 5 = call_action_queue
busy_type	Tinyint(3)	Event “on busy”, type of entity to be shown: 0 = callable_user 1 = callable_group 2 = callable_gateway 3 = callable_ivr 4 = callable_service 5 = callable_lcr 6 = callable_messages 7 = callable_device 8 = callable_wan_user 9 = callable_voicemail 10 = callable_voicemail_manager
busy_id	int(10)	Event “on busy”: show ID to show
busy_num	Varchar(20)	Event “on busy”: number to call
busy_msg	Tinyint(3)	Event “on busy”, type message: 0 = audio_none 1 = audio_file 2 = audio_linein 3 = audio_dynamic Event “on busy”,
busy_src	Varchar(255)	Event “on busy”, message source
unre_act	Tinyint(3)	Event “on unreachable”, action to be taken: (see busy_act)
unre_type	Tinyint(3)	Event “on unreachable”, type of entity to be shown (see busy_type)
unre_id	int(10)	Event “on unreachable”, ID to be shown
unre_num	Varchar(20)	Event “on unreachable”, number to call
unre_msg	Tinyint(3)	Event “on unreachable”, type message: (see busy_msg)
unre_src	Varchar(255)	Event “on unreachable”, message source
maxr_act	Tinyint(3)	Event “on maximum ring time”, action to be taken: (see busy_act)
maxr_type	Tinyint(3)	Event “on maximum ring time”, type of entity to be shown: (see busy_type)
maxr_id	int(10)	Event “on maximum ring time”, ID to be shown
maxr_num	Varchar(20)	Event “on maximum ring time”, number to call
maxr_msg	Tinyint(3)	Event “on maximum ring time”, type message: (see busy_msg)
maxr_src	Varchar(255)	Event “on maximum ring time”, message source

Manual S.S.N. (System Status Notification)

PBX Configuration

VOIspeed PBX can send notification content with all the information on actions performed, for example, on incoming and outgoing calls, responding etc. The process is made via a GET http/https which comprises various settings.

To activate the notification, you need to configure certain settings in Configuration → Company → Settings, where you need to insert "URL SSN Script", or the URL from the script produced by VOIspeed, indicating which events you wish to receive.

So as not to overload the receiving server with all the events from the phone system, the integration is automatically active only for calls activated via the SOAP command CallReqExt.

If you need to activate the integration for every call event, including from the phone or GUI, you need to explicitly enable it from the appropriate routing rules. For example, to receive call events from or to a gateway, you need to enable the integration in the rule linked to that gateway.

We should stress that some entities (users or service numbers) do not require routing rules to be called, as the system configures rules automatically. If you wish to receive events on one of these entities you need to create an ad-hoc rule and activate integration.

Introduction to calls

From a high level, a call in VOIspeed is called "Commutation" identified by the "call_id" disconnected in all S.S.N. events.

Each Commutation always has 2 channels:

- Calling channel identified by
  - calling_id
  - calling_type
  - calling_local_id (identified by the branch of the call from the channel calling)
- Called channel identified by:
  - called_id
  - called_type
  - called_local_id (identified by the branch of the call from the channel called)

NB: calling_local_id and called_local_id are:

- Identified by part of the calls respectively from the calling and called channels
- Not to be confused with the general switch identification "call_id"
- Only disconnected when the SSN 6 event "Call response" has a positive outcome. Each time part of a call is accepted then the event 6 is created with an updated summary of the state of both commutation channels

Documentation for events and their settings

The following events represent the call events manged by the system, specifically those that use the PBX itself as an observer. In the following paragraph they will be distinguished from those instead, that have the user as a point of reference.

I seguenti eventi rappresentano gli eventi di chiamata gestiti dal sistema, ossia che prendono il PBX stesso come osservatore. Nel successivo paragrafo li distingueremo da quelli che hanno, invece, l'utente come punto di riferimento.

1. System Events

Every S.S.N. event is a GET http towards a script (which is produced and uploaded by the system integrator) specified in the space on the system configurator in the section: Configuration→Company →Settings.

1. Every S.S.N. event contains the opening following parameters:
    - **pbx_id**= identified from the pbx specified in the field "c_idpbx"
    - **key**=event counter which keeps track of the order of events
    - **event**=event code (details are given below)
    - **datetime**=date and time of the event; format=YYYY.MM.DD.HH.MM.SS.mmm
    - **domain**= company domain

2. Events:

    - 3 = incoming call
        - call_id=global call identifier
        - call_id_ext=external identifier
        - call_status=see call status values (4)
        - calling= calling number
        - called= called number
        - calling_id= identifier of calling entity
        - calling_type=type of calling entity; see entity type values (5)

    - 4 = call progress
        - call_id=global call identifier
        - call_id_ext=external identifier
        - call_status=see call status values (4)
        - sender_id= entity identifier of progress signal sender
        - sender_type = entity type of progress signal sender; see entity type values (5)
        - progress_code= code of progress signal (180 ringing\183 progress)

    - 5 = outgoing call
        - call_id=global call identifier
        - call_id_ext=external identifier
        - call_status=see call status values (4)
        - calling=calling number
        - called= called number
        - called_id=identifier of called entity
        - called_type=type of called entity; see entity type values (5)

    - 6 = call response
        - call_id=global call identifier

- call_id_ext=external identifier
- call_status=see call status values (4)
- result=response outcome; see call outcome values (6)
- sender_id= entity identifier of response sender
- sender_type = entity type of response sender; see entity type values (5)
- connected_id= identifier of connected entity
- connected_type=type of connected entity; see entity type values (5)
- calling_id= identifier of calling channel entity
- calling_type=type of calling channel entity; see entity type values (5)
- calling_local_id=identifier of calling channel call
- called_id=identifier of called channel entity
- called_type= type of called channel entity; see entity type values (5)
- called_local_id= identifier of called channel call
- reject_code= SIP code response (according to RFC 3261)

- 7 = incoming disconnect
  - call_id=global call identifier
  - call_id_ext=external identifier
  - call_status=see call status values (4)
  - sender_id= entity identifier of disconnect sender
  - sender_type = entity type of disconnect sender; see entity type values (5)

- 8 = outgoing disconnect
  - call_id=global call identifier
  - call_id_ext=external identifier
  - call_status=see call status values (4)
  - recipient_id= identifier of disconnect entity recipient
  - recipient_type = type of disconnect entity recipient; see entity type values (5)

- 9 = consultative call transfer
  - call_id=global call identifier
  - call_id_ext=external identifier
  - call_status=see call status values (4)
  - sender_id= entity identifier of transfer request sender
  - sender_type = entity type of transfer request sender; see entity type values (5)
  - refer_id: call identifier of transferred party
  - recipient_id: entity identifier of transfer request recipient
  - recipient_type: entity type of transfer request recipient; see entity type values (5)

- 10 = dtmf
  - call_id=global call identifier
  - call_id_ext=external identifier
  - call_status=see call status values (4)
  - sender_id= entity identifier of dtmf sender
  - sender_type = entity type of dtmf sender; see entity type values (5)
  - info: dtmf code

- 11 = call divert
  - call_id=global call identifier
  - call_id_ext=external identifier
  - call_status=see call status values (4)

- sender_id= entity identifier of divert request sender
- sender_type = entity type of divert request sender; see entity type values (5)
- recipient_id= identifier of divert entity recipient
- recipient_type = type of divert entity recipient; see entity type values (5)
- recipient_number= called number to be send to recipient entity

- 12 = call terminated
  - call_id=global call identifier
  - call_id_ext=external identifier
  - call_status=see call status values (4)
  - outcome=see Call outcome values (6)

- 13 = entity status change
  - sender_id= identifier of entity that changed status
  - sender_type = type of entity that changed status; see entity type values (5)
  - number = number of entity that changed status (valid information only for entity user)
  - status = audio video entity status; see entity status value (7)
  - gui_status= gui entity status (valid information only for entity user); see entity status value (7)
  - num_calls= number of calls where the entity is currently involved

- 14 = call held
  - call_id=global call identifier
  - call_id_ext=external identifier
  - call_status=see call status values (4)
  - user_id= identifier of user that put the call on hold
  - user_call_id= identifier of the user call put on hold

- 15 = call retrieved
  - call_id=global call identifier
  - call_id_ext=external identifier
  - call_status=see call status values (4)
  - user_id= identifier of user that retrieved the call
  - user_call_id= identifier of the user call retrieved

3. Values for the call status:
   - 0 = idle
   - 1 = incoming
   - 2 = calling
   - 3 = active
   - 4 = diverting
   - 5 = picking up
   - 6 = rejecting
   - 7 = terminating
   - 8 = terminated

4. Values for the entity:
   - 0 = user (users table)
   - 1 = group (groups table)

- 2 = gateway (gateways table)
- 3 = ivr (ivr table)
- 4 = service (no related table)
- 5 = lcr (no related table)
- 6 = system message
- 7 = device (monitor_devices table)
- 8 = wan user (wan_users table)
- 9 = company voicemail (voicemails table)
- 11 = company voicemail manager (voicemails table)
- 12 = meeting room (meeting_room table)
- 13 = park line (park_line table)

5. Values for the call outcome:
    - 0 = undefined
    - 1 = ok
    - 2 = busy
    - 3 = rejected
    - 4 = unreachable
    - 5 = no answer
    - 6 = no free resources
    - 7 = dnd (you have this outcome only when the recipient entity is a user in do not disturb mode)
    - 8 = absent (you have this outcome only when the recipient entity is a user in do absent mode)
    - 9 = calling timeout (you have this outcome only when you have specified calling_timeout parameter into call_request_ext command)

6. Values of the entity status:
    - 0 = unknown
    - 1 = unreachable
    - 2 = available
    - 3 = ringing
    - 4 = busy
    - 5 = do not disturb
    - 6 = not used
    - 7 = absent

2. User Events

Below are the details of the status and call events that the PBX sends to the same script, from the user's point of view.

In practice each event specifies a user as a subject (in a specific parameter for each event sent) and the company to which they are linked.

1. Every S.S.N event is a GET http towards the script specified in the field of the company "integration URL script".
2. Every S.S.N. event has the following parameters to begin with:
    - pbx_id= pbx identifier specified into company field "c_idpbx"
    - key=event counter
    - event=event code
    - datetime=date and time; as follows=YYYYMMDDHHMMSSmmm
    - domain= company domain
    - company=Company ID (referenced in table XXXX)

3. There are also specific parameters for each event that show the following details event by event::
    - ssn_event = 1 a user device has logged in

      device_id = Contains the ID of the device that has logged in referenced to the table monitor_devices

      device_type = Contains the type of device that has logged in and assumes the following values:
      - 1 = gui
      - 2 = sip phone
      - 3 = usb device
      - 4 = dect device
      - 5 = mobile

      device_host = Contains IP and port of the device that has logged in in the format address_ip:port

      device_name = Contains the specific user agent for every device

      number = Contains the extension of the user on which the device has logged in

      user_id = Contains the user identification, referenced in the field "ID" from the "users" table

    - ssn_event = 2 a user device has logged out

      device_id = Contains the ID of the device that has logged out, referenced in the table monitor_devices

      device_type = Contains the type of device that has logged out and with these values:
      - 1 = gui
      - 2 = sip phone
      - 3 = usb device
      - 4 = dect device
      - 5 = mobile

      device_host = Contains the IP and port of the device that logged out in the format address_ip:port

device_name = Contains the specific user agent for every device

number = Contains the user extension for the device that logged out

user_id = Contains the user identification, referenced in the field "ID" of the "users" table

- ssn_event = 16 shows the arrival of a call for a given user. The event contains the following settings:

  sender_id and sender_type = Contains the ID and type of user respectively; the type can assume, in this event, only the value 0 (when 0 is the type of user).

  call_id = Containing the unique call ID allowing tracking in the PBX reporting tables (CAUTION - this is not the unique calling call_ID but the user_call_ID, which can be traced in the user_calls table)

  ext_id = Containing the call identifier (useful setting for external applications and generated by the user on demand from these same applications).

  number = Contains the calling number

  ext = Contains the number called

  identity = Containing the caller's identity in a string with the following format: Name|Surname|Department|parameter1_component |parameter2_component_name (in particular the two most common parameters, outside the Polyedro world, will always be set to 0)

  group = Containing the name of the group to which the user called belongs, in case the call arrives as belonging to the group.

- ssn_event = 17 indicates the exit of a call towards a given user. The event contains the following parameters:

  sender_id e sender_type = Contains the sender id and type of event sender: the *type* can assume, in this event, only the value 0 (since 0 is the type of user).
  call_id = Contains the unique ID of the call which allows tracking in the PBX reporting table (CAUTION - this is not the unique call_ID of the call but the user_call_ID, which can be traced in the user_calls table)

  ext_id = Contains the relative call identifier (parameter useful for external applications and generated by the user at the request of the applications themselves)
  <number></number>
  Contains the recipient's number of the outgoing call

  ext = Containing the calling number

identity = Containing the identity of the recipient of the call in a string with the following format
Name|Surname|Department|parameter1_component|
parameter2_component_name (in particular, the last two parameters, outside the Polyedro world, will always be set to 0)

group = Containing the name of the group to which the user called belongs, in case the call arrives as belonging to the group.

ssn_event = 18 indicates the user_call_response

- if the call begins with the event 16 (user incoming call) the event is triggered when the call is accepted by the user
- if the call begins with the event 17 (user outgoing call) the event is triggered when the number called by the user has accepted the call.

The event contains the following parameters:

sender_id and sender_type = Contains the user and type of event sender: the *type* can assume, in this event, only the value 0 (since 0 is the type of user).

and the *ID* is the *ID* of the element present in the reference table of the type.

call_id = Containing the unique call ID allowing tracking in the PBX reporting tables (CAUTION - this is not the unique calling call_ID but the user_call_ID, which can be traced in the user_calls table)

ext_id = Contains the relative call identifier (parameter useful for external applications and generated by the user at the request of the applications themselves)

number = Contains the recipient's number of the outgoing call (the one who answers)

ext = Containing the calling number

identity = Containing the identity of the recipient of the call in a string with the following format:
Name|Surname|Department|parameter1_component|
parameter2_component_name (in particular, the last two parameters, outside the Polyedro world, will always be set to 0)

group = Containing the name of the group to which the called user belongs, in case the call arrives as belonging to the group.

.

ssn_event = 19 indicates the user_call_reject

- if the call began with event 16 it means that the user rejected it

- if the call began with event 17 it means the number called by the user rejected their call.

sender_id and sender_type = Contains the user and type of event sender: the *type* can assume, in this event, only the value 0 (since 0 is the type of user).

call_id = Containing the unique call ID allowing tracking in the PBX reporting tables (CAUTION - this is not the unique calling call_ID but the user_call_ID, which can be traced in the user_calls table)

ext_id = Contains the relative call identifier (parameter useful for external applications and generated by the user at the request of the applications themselves)

number = Contains the destination number of the outgoing call (of the responder)

ext = Containing the calling number

identity = Containing the identity of the recipient of the call in a string with the following format
Name|Surname|Department|parameter1_component|
parameter2_component_name (in particular the last two parameters, outside the Polyedro world, they will always be set to 0)

group = Containing the name of the group to which the called user belongs, in case the call arrives as belonging to the group.

- ssn_event = 20 l'interlocutore dell'utente ha chiuso la chiamata. L'evento contiene i seguenti parametri:

sender_id and sender_type = Contains the user and type of event sender: the *type* can assume, in this event, only the value 0 (since 0 is the type of user).

call_id = Containing the unique call ID allowing tracking in the PBX reporting tables (CAUTION - this is not the unique calling call_ID but the user_call_ID, which can be traced in the user_calls table)

ext_id = Containing the relative call identifier (parameter useful for external applications and generated by the user at the request of the applications themselves)

number = Containing the number of the one who sends the disconnection

ext = Contains the number that receives the disconnection

identity = Contains the identity of the call disconnect sender in a string with the following format:

Name | Surname | Department | Parameter1_component_name | Parameter2_component_name (in particular the last two parameters, outside the Polyedro world, will always be set to 0)

group = Containing the name of the group to which the called user belongs, in case the call arrives as belonging to the group.

- ssn_event = 21 the user has closed the call. The event contains the following parameters:

sender_id and sender_type = Contains the user and type of event sender: the *type* can assume, in this event, only the value 0 (since 0 is the type of user).

call_id = Containing the unique call ID allowing tracking in the PBX reporting tables (CAUTION - this is not the unique calling call_ID but the user_call_ID, which can be traced in the user_calls table)

ext_id = Containing the relative call identifier (parameter useful for external applications and generated by the user at the request of the applications themselves)

number = Containing the number of the entity that will receive the disconnection

ext = Containing the number of the entity that sends the disconnection

identity = Containing the identity of the recipient of the call in a string with the following format:

Name | Surname | Department | Parameter1_component_name | Parameter2_component_name (in particular the last two parameters, outside the Polyedro world, will always be set to 0)

group = Containing the name of the group to which the called user belongs, in case the call arrives as belonging to the group.