CloudPBX API

Poziom7

2023/12/22 12:15

Spis treści

CloudPBX API

Dokumentacja interfejsu systemu CloudPBX

Wersja 1.1

System CloudPBX udostępnia zestaw narzędzi do prostego zarządzania wirtualną centralą telefoniczną poprzez aplikacje webowe. W szczególnych przypadkach istnieje potrzeba integracji systemu telekomunikacyjnego z systemem informatycznym klienta w zakresie pobierania danych połączeń telefonicznych. Funkcjonalność taką umożliwia udostępnione publiczne API CloudPBX oparte o interfejsy REST/WebSocket

1. Komunikacja REST

Interface API w zakresie pobierania danych oparty jest o specyfikację REST, protokół HTTPS i format JSON. Wymiana komunikacji odbywa się w szyfrowanym kanale SSL protokołu HTTP.

Adresy interface CloudPBX:

https://api.cloudpbx.pl, https://api2.cloudpbx.pl

1.1 Autoryzacja

Autoryzacja i autentykacja odbywa się za pomocą klucza API, przekazywanego przez dostawcę usługi. Klucz musi zostać umieszczony w nagłówku HTTP

```
Authorization: Bearer [AUTHKEY]
```

1.2 cdr

Polecenie *cdr* pozwala na pobranie listy rekordów billingowych za dany okres. Do pobrania danych wykorzystywana jest metoda GET.

Przykład wywołania CURL:

```
curl -X GET -H 'Authorization: Bearer [APIKEY]' https://api.cloudpbx.pl/cdr
```

Standardowo wywołanie zwraca listę rekordów billingowych w bieżącym dniu. Aby zmienić zakres dat należy podać parametry 'datestart' i 'datestop' w formacie „YYYY-MM-DD”

```
curl -X GET -H 'Authorization: Bearer [APIKEY]'
https://api.cloudpbx.pl/cdr?datestart=2021-10-01&datestop=2021-10-07
```

Polecenie zwraca tablicę z listą rekordów w formacie JSON:


```
[
  {
    "calldate":"2021-12-15 11:25:02",
    "clid":"\"Sekretariat 3\" <101>",
    "src":"101",
    "dst":"105",
    "channel":"SIP\/PBX-101-00000028",
    "dstchannel":"SIP\/PBX-105-00000029",
    "lastapp":"Queue",
    "duration":13,
    "billsec":13,
    "disposition":"ANSWERED",
    "uniqueid":"1639563902.75",
    "cost":"",
    "zone":""
  }
]
```


Pola rekordu:

- calldate - czas rozpoczęcia połączenia
- clid - prezentacja numeru dzwoniącego
- src - numer źródłowy
- dst - numer dzwoniony
- channel - kanał urządzenia rozpoczynającego połączenie
- dstchannel - kanał terminujący połączenie w centrali
- lastapp - aplikacja zestawiająca połączenie
- duration - czas połączenia łącznie z czasem zestawiania
- billsec - czas połączenia po zgłoszeniu drugiej strony
- disposition - stan finalny połączenia: ANSWERED, NO ANSWER, BUSY, FAILED
- uniqueid - identyfikator połączenia
- cost - koszt połączenia podany w groszach
- zone - strefa billingowa połączenia

Uwaga! metoda pozwala na pobranie maksymalnie 10 000 rekordów CDR

1.3 record

Polecenie *record* pozwala na pobranie listy nagrań w danym okresie lub konkretnego nagrania do zapisu na dysku lokalnym. Do pobrania danych wykorzystywana jest metoda GET.

```
curl -X GET -H 'Authorization: Bearer [APIKEY]'
https://api.cloudpbx.pl/record
```

Wywołanie polecenia bez parametrów powoduje zwrócenie listy nagrań w formacie JSON. Domyślnie polecenie zwraca nagrania z bieżącego dnia. Aby zmienić zakres dat należy podać parametry 'datestart' i 'datestop' w formacie „YYYY-MM-DD”

```
curl -X GET -H 'Authorization: Bearer [APIKEY]'
https://api.cloudpbx.pl/record?datestart=2021-10-01&datestop=2021-10-07
```

Polecenie zwraca tablicę obiektów JSON:

```
[
  {
    "uniqueid":"1552558647.371",
    "src":"101",
    "dst":"105",
    "format":"wav",
    "date":"2019-03-14 11:17:28"
  }
]
```

Pola struktury:

- uniqueid - identyfikator połączenia
- src - numer źródłowy
- dst - numer docelowy
- format - format nagrania
- date - data i godzina połączenia

Uwaga! polecenie pozwala na pobranie listy maksymalnie 5000 nagrań

Pobranie nagrania:

```
curl -X GET --output record.wav -H 'Authorization: Bearer [APIKEY]'
https://api.cloudpbx.pl/record/[uniqueid]
```

Wywołanie polecenia *record* z podaniem identyfkatora powoduje pobranie nagrania. W przypadku CURL warto dodać opcję –output pozwalająca na zapis danych do pliku.

1.4 dial

Polecenie *dial* pozwala na wywołanie połączenia pomiędzy abonentem A i abonentem B. Do wykonania połączenia wykorzystywana jest metoda GET.

Polecenie wymagapodania parametrów

- cpa - numer źródłowy w obrębie centrali
- cpb - numer docelowy

Przykład wywołania CURL:

```
curl -X GET -H 'Authorization: Bearer [APIKEY]'
https://api.cloudpbx.pl/dial?cpa=101&cpb=200
```

Wywołanie polecenia *dial* zwraca

- OK [kod 200] w przypadku powodzenia
- Action Error [kod 500] w przypadku błędu wywołania

1.5 sendSMS

Polecenie dostępne jedynie w przypadku, gdy centrala ma udostępnioną bramkę SMS. Aby aktywować usługę należy skontaktować się z biurem obsługi klienta CloudPBX.

Polecenie *sendSMS* pozwala na wysłanie wiadomości tekstowej do sieci GSM poprzez bramkę SMS. Do wysłania wykorzystana jest metoda POST, w której należy podać paramerty wadomości:

- to - dziewięciocyfrowy numer telefonu w sieci komórkowej, do którego kierowana jest wiadomość
- body - tekst wiadomości
- test - parametr opcjonalny. Jeśli jest ustawiony i jego wartość zawiera string „true”, wiadomość traktowana jest jako testowa - bez dostarczania do odbiorcy i naliczania kosztu.

UWAGA! maksymalna długość wiadomości wynosi 1024 znaki.

Przykład wywołania CURL:

```
curl -v -X POST -H 'Authorization: Bearer [APIKEY]'\
 -d "to"="[GSMNUMBER]"\
 -d "body"="tekst wysyłanej wiadomości"\
 -d "test"="true"\
   https://api.cloudpbx.pl/sendSMS
```

Gdzie:

- GSMNUMBER - dziewięciocyfrowy numer w sieci GSM

Wywołanie polecenia zwraca

- RESPONSE [kod 200] w przypadku powodzenia
- RESPONSE [kod 500] w przypadku błędu wywołania
- Parameter is missing [kod 400] w przypadku braku lub niepoprawnych parametrów „to” lub „body”

Gdzie: RESPONSE to obiekt w formacie json. Przykładowe wartości obiektu RESPONSE:

```
{"responseId":"3421756012963258480","responseStatus":"QUEUE","responseErrorM
esage":null}
```

Wartość pola *responseId* określa unikalny numer identyfikacyjny wiadomości, dzięki temu pozwala na monitorowanie stanu wysyłki wiadomości. Aktualny stan przesyłany jest kanałem zwrotnym komunikacją WS Po zmianie stanu, w kanale WS przesyłany jest komunikat smsSendResponse zawierający aktualne informacje o wysyłce.

Wartości pola *responseStatus* opisane są w punkcie 2.3.5 (smsSendResponse).

Pole *responseErrorMesage* zawiera wiadomość błędu jeśli taki wystąpił. Wartość null oznacza poprawne wykonanie komendy oraz przekazanie wiadomości SMS do wysyłki.

Dodatkowo *responseId* zapisywane jest w rekordzie CDR w polu *uniqueid*.Dzięki temu można odszukać rekord cdr zawierający parametry taryfikacyjne w tabeli billingów.

2. Komunikacja WebSocket

Interface WebSocket jest mechanizmem działającym odmiennie od REST. O ile REST zwraca odpowiedzi na zadane pytania, o tyle WebSocket strumieniuje zdarzenia do klienta. Po podłączeniu klienta do WebSocket, na bieżąco przesyłane są komunikaty o zmianie stanów centrali i urządzeń. Dzięki temu kient może na bieżąco reagować na przychodzące zdażenia.

Protokół WebSocket opiera się częściowo na standardzie HTTP, wykorzystując nagłówek UPGRADE, który powoduje otwarcie kanału komunikacyjnego, którym przesyłane są komunikaty z serwera. WebSocket wspierany jest przez współczesne przeglądarki i JS. Nie brakuje również bibliotek w wielu językach programowania pozwalających łatwo zaimplementować obsługę WS.

Najprostsza imlementacja może opierać się na aplikacji curl lub websocat (https://github.com/vi/websocat). Podłączenie do websocketu umożliwia podgląd przychodzących

komunikatów.

Cześć języków zawierających wbudowaną obsługę protokołu HTTP umożliwia wykorzystanie WS poprzez odpowiednią konstrukcję nagłówka HTTP oraz wykorzystanie metody UPGRADE. Niektóre języki (JS, C#) posiadają wbudowane biblioteki wspomagające obsługę WS, inne oferują zewnętzne biblioteki:

- PHP: textalk/websocket (https://github.com/Textalk/websocket-php)
- GoLang: gorilla/websocket (https://github.com/gorilla/websocket)

2.1 Nawiązanie połączenia WS

Do nawiązania połączenia z api CloudPBX niezbędne jest posiadanie adresu serwera API, nazwy centrali oraz klucza autoryzacyjnego. Te dane pozwolą na zbudowanie prawidłowego URL:

```
[PROTO]://[SERVER]/ws/pbx_[PBX_NAME]?pbxkey=[APIKEY]
```

gdzie:

- PROTO: nazwa protokołu http/https lub ws/wss
- SERVER: adres serwera api
- PBX_NAME: nazwa centrali, której eventy zostaną przechwycone
- APIKEY: klucz niezbędny do autoryzacji

Przykładowe podłączenie do WebSocket za pomocą CURL:

```
curl --include \
     --no-buffer \
     --output - \
     --header "Connection: Upgrade" \
     --header "Upgrade: websocket" \
     --header "Host: pbxvisor1.cloudpbx.pl" \
     --header "Origin: https://pbxvisor1.cloudpbx.pl" \
     --header "Sec-WebSocket-Key: SGVsbG8sIHdvcmxkIQ==" \
     --header "Sec-WebSocket-Version: 13" \
     https://pbxvisor1.cloudpbx.pl/ws/pbx_MojaCentrala?pbxkey=MyExampleKey
```

Przykładowe połączenie za pomocą aplikacji websocat

```
websocat
https://pbxvisor1.cloudpbx.pl/ws/pbx_MojaCentrala?pbxkey=MyExampleKey
```

2.2 Komunikaty centrali

Po podłączeniu do portu WebSocket w momencie wystąpienia zdarzenia na centrali zostaje wygenerowany event. Eventy podawane są w formacje JSON w poniższym formacie:

```
{"id":386,"channel":"pbx_MojaCentrala","text":{data}}
```

gdzie

- id: kolejny numer komunikatu
- channel: nazwa kanału centrali
- text: zawiera jeden lub więcej obiektów zdarzeń

Przychodzący komunikat zawiera jeden lub więcej eventów zawierających dodatkowe informacje dotyczące każdego zdarzenia.

2.3 Opis eventów

2.3.1 subscriber

```
{
"subscriber":
    {
    "peer":"MojaCentrala-101",
    "statusValue":1
    }
}
```

Event **subscriber** opisuje stan konta abonenckiego (peera). opis parametrów eventu:

- peer - opisuje konto abonenckie, którego nazwa składa się z nazwy centrali oraz numeru abonenckiego
- statusValue - opisuje stan peera. Wartości stanów podane są w punkcie 2.4
- connectPeerNumber - pole nieobowiązkowe, zawiera numer abonenta w przypadku eventu opisującego połączenie
- connectPeerName - pole nieobowiązkowe, zawiera prezentację abonenta w przypadku eventu opisującego połączenie

2.3.2 extstatus

```
{
"extstatus":
    {
    "exten":"130",
    "status":4
    }
}
```

Event **extstatus** opisuje stan aparatu. Często jest on równoznaczny ze stanem eventu subscriber, opis parametrów eventu:

- exten - numer abonencki aparatu
- statusValue - stan aparatu. Wartości stanów podane są w punkcie 2.4

2.3.3 queue

```
{
"queue":
    {
    "peer":"Kolejka1",
    "calls":"1"
    }
}
```

Event **queue** opisuje zmianę stanu w kolejce. Gdy w danej kolejce zmienia się liczba oczekujących połączeń, ich ilość raportowana jest za pomocą tego zdarzenia.

- peer - nazwa kolejki
- calls - liczba połączeń oczekujących w kolejce

2.3.4 qmember

```
{
"qmember":
    {
    "peer":"MojaCentrala-101",
    "queue":"Kolejka1",
    "m_status":"1",
    "m_ct":"0",
    "m_mbshp":"static",
    "m_paused":"0",
    "m_name":"Agent 1",
    "m_location":"SIP\/MojaCentrala-101"
    }
}
```

Event **qmember** generowany jest w przypadku zmiany stanu agenta obsługującego kolejkę

- peer - nazwa konta agenta którego nazwa składa się z nazwy centrali oraz numeru abonenckiego
- queue - nazwa kolejki, w której zdarzenie dotyczy agenta
- m_status - stan agenta w kolejce. Wartości stanu agentów podane są w punkcie 2.4
- m_ct - ilość odebranych połączeń przez agenta
- m_mbshp - rodzaj agenta - „dynamic” - agent dynamiczny, „static” - agent statyczny
- m_paused - wartość 1 oznacza agenta w trakcie pauzy (nie odbiera połączeń), wartość 0 oznacza agenta gotowego do pracy
- m_reason - jeśli event dotyczy zmiany stanu pauzy to pole może zawierać powód zmiany stanu
- m_name - nazwa agenta
- m_location - identyfikator terminala który odbiera połączenia

2.3.5 cmember

```
{
"cmember":
    {
    "channel":"SIP\/MojaCentrala-101-0005f005",
    "action":"add",
    "conference":"SesjaRady",
    "callerIDName":"Jan Nowak",
    "callerIDNum":"101"
    }
}
```

Event **cmember** opisuje zmianę stanu uczestnika pokoju konferencyjnego

- channel - nazwa kanału akustycznego zawierający urządzenie końcowe
- action - „add” kiedy użytkownik dołącza do pokoju, remove kiedy opuszcza pokój
- conference - nazwa pokoju konferencyjnego
- callerIDName - prezentacja użytkownika konferencj
- callerIDNum - numer abonencki użytkownika konferencji

2.3.5 smsSendResponse

```
{
"smsSendResponse":
     {
     "MsgId": "3292468348841377230",
     "to": "48XXXXXXXXX",
     "status": "SENT"
     }
}
```

Event **smsSendResponse** przesyłany jest w momencie zmiany stanu zadania wiadomości SMS wysłanej do sieci komórkowej za pomocą bramki cloudPBX.

- MsgID - unikalny numer wiadomości przyznawany podczas wysyłania
- to - numer w formacie międzynarodowym, do którego została wysłana wiadomość
- status - aktualny stan zadania

Możliwe raportowane stany

- NOT_FOUND - Nieznaleziona. Błędny numer ID lub raport wygasł
- EXPIRED - Przedawniona. Wiadomość niedostarczona z powodu zbyt długiego czasu niedostępność numeru
- SENT - Wysłana. Wiadomość została wysłana ale operator nie zwrócił jeszcze raportu doręczenia
- DELIVERED - Dostarczona. Wiadomość dotarła do odbiorcy
- UNDELIVERED - Niedostarczona. Wiadomość niedostarczona (np.: błędny numer, numer niedostępny)
- FAILED - Nieudana. Błąd podczas wysyłki wiadomości - prosimy zgłosić
- REJECTED - Odrzucona. Wiadomość niedostarczona (np.: błędny numer, numer niedostępny)
- UNKNOWN - Nieznany. Brak raportu doręczenia dla wiadomości (wiadomość doręczona lub brak możliwości doręczenia)

- QUEUE - Kolejka. Wiadomość czeka w kolejce na wysyłkę
- ACCEPTED - Zaakceptowana. Wiadomość przyjęta przez operatora
- RENEWAL - Ponawianie. Wykonana była próba połączenia która nie została odebrana, połączenie zostanie ponowione.
- STOP - Zatrzymanie.

2.4 Wartości stanów

Pole statusValue eventu subscriber przyjmuje wartości liczbowe określające stan konta abonenckiego:

- 0 - „Wolny” (Idle)
- 1 - „Rozmawia” (In Use)
- 2 - „Zajęty” (Busy)
- 4 - „Niedostępny” (Unavailable)
- 8 - „Dzwoni” (Ringing)
- 16 - „Zawieszony” (On Hold)
- 32 - „Łączenie” (Connecting)

Pole m_status eventu qmember przyjmuje wartości liczbowe określające stan agenta:

- 0 - Nieznany
- 1 - Wolny
- 2 - Rozmawia
- 3 - Zajęty
- 4 - Nierawidłowy
- 5 - Niedostępny
- 6 - Dzwoni
- 7 - Dzwoni przy zajętości
- 8 - Zawieszony

3. Przykładowy kod kliencki

W celu testów i własnej implementacji, pod adresem https://github.com/grzegorziwaniec/cloudpbx_api_client został udostępniony przykładowy kod klienta w języku PHP. Kod prezentuje metody autoryzacji oraz sposób wykorzystania dostępnych metod API zarówno protokołu REST jaki WebSocket

Wygenerowane na podstawie:
https://wiki.poziom7.pl/ - **Poziom7**

Bezpośredni link:
https://wiki.poziom7.pl/doku.php?id=cloudpbx:apps:cloudpbx_api

Ostatnia aktualizacja: **2023/12/22 12:15**