Инструкция по подключению FreePBX к облачному сервису системы статистики FETG.UZ

Вводная

- Данная инструкция написана, используя FreePBX версии 13.0.191.11 с Asterisk 13.14.0 на борту.
  При возникновении каких-либо сложностей/вопросов с более ранними/поздними версиями системы, просьба написать нам на support@fetg.uz.
- Для выполнения описанных ниже действий понадобится:
  - умение подключиться к серверу по ssh
  - умение редактировать файлы в текстовом редакторе (nano/vim/emacs/etc)
  - ясная голова и хорошее настроение 😉

Настройка

В поставке по умолчанию (что называется «из коробки») FreePBX ведет БД asteriskcdrdb, а точнее табличку cdr, не совсем так, как это нужно для правильной работы системы статистики FETG.UZ. Также она совсем не пишет в таблицу queue_log, что является необходимым. Поэтому нам потребуется выполнить несколько шагов настройки, для достижения нужного поведения FreePBX.

1. Начнем с таблицы cdr.
   Подключимся к серверу по ssh, а затем к консоли MySQL-сервера

```
[root@localhost ~]# mysql asteriskcdrdb
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 1686
Server version: 5.1.73 Source distribution

Copyright (c) 2000, 2013, Oracle and/or its affiliates. All rights
reserved.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input
statement.
```

2. Добавим табличке cdr новые поля id и filename и повесим триггер на INSERT для заполнения filename

```
ALTER TABLE cdr ADD COLUMN filename VARCHAR(120) DEFAULT 'none';
UPDATE cdr SET filename=recordingfile;
ALTER TABLE cdr ADD COLUMN id INT(11) AUTO_INCREMENT PRIMARY KEY;
DELIMITER $$
CREATE TRIGGER `before_insert_cdr` BEFORE INSERT ON `cdr` FOR EACH ROW
```

```
BEGIN
SET NEW.filename=NEW.recordingfile;
END $$
DELIMITER ;
```

При успешном выполнении запросов все должно выглядеть примерно вот так:

3. Совершим вызов и проверим, что поле filename заполняется
4. Теперь нам необходимо добавить два новых поля в таблицу cdr - realdst и realsrc, в которые будут записываться номера звонящего и назначения (например, не номер оператора очереди, а наш городской номер, на который пришел вызов извне для realdst или не номер транка, а номер внутреннего абонента для realsrc).
   Для этого выполним sql-запросы

```
ALTER TABLE `cdr` ADD `realsrc` VARCHAR(80) NOT NULL DEFAULT '';
ALTER TABLE `cdr` ADD `realdst` VARCHAR(80) NOT NULL DEFAULT '';
UPDATE cdr SET realsrc=src;
UPDATE cdr SET realdst=dst;
```

5. Пришло время последнего, но отнюдь не маловажного шага в разрезе конфигурирования таблицы cdr - научим АТС записывать данные во вновь созданные поля.

   Отредактируйте /etc/asterisk/extensions_override_freepbx.conf, внеся в него

```
[ext-did-catchall]
include => ext-did-catchall-custom
exten => _.,1,Noop(Catch-All DID Match - Found ${EXTEN} - You probably
want a DID for this.)
exten => _.,n,Set(__FROM_DID=${EXTEN})
exten => _.,n,Set(_FETGUZREALDST=${FROM_DID})
exten => _.,n,Set(_FETGUZREALSRC=${CALLERID(num)})
exten => _.,n,Set(CDR(realsrc)=${FETGUZREALSRC})
exten => _.,n,Set(CDR(realdst)=${FETGUZREALDST})
exten => _.,n,Goto(ext-did,s,1)


[sub-record-check]
include => sub-record-check-custom
exten => s,1,GotoIf($[${LEN(${FROMEXTEN})}]?initialized)
exten => s,n,Set(__REC_STATUS=INITIALIZED)
exten => s,n,Set(NOW=${EPOCH})
exten => s,n,Set(__DAY=${STRFTIME(${NOW},,%d)})
exten => s,n,Set(__MONTH=${STRFTIME(${NOW},,%m)})
exten => s,n,Set(__YEAR=${STRFTIME(${NOW},,%Y)})
exten => s,n,Set(__TIMESTR=${YEAR}${MONTH}${DAY}-
${STRFTIME(${NOW},,%H%M%S)})
exten =>
s,n,Set(__FROMEXTEN=${IF($[${LEN(${AMPUSER})}]?${AMPUSER}:${IF($[${LEN(
${REALCALLERIDNUM})}]?${REALCALLERIDNUM}:unknown)})})
exten =>
```

```
s,n,Set(__MON_FMT=${IF($["${MIXMON_FORMAT}"="wav49"]?WAV:${MIXMON_FORMA
T})})
exten => s,n(initialized),Noop(Recordings initialized)
exten => s,n,ExecIf($[!${LEN(${ARG3})}]?Set(ARG3=dontcare))
exten => s,n,Set(REC_POLICY_MODE_SAVE=${REC_POLICY_MODE})
exten => s,n,ExecIf($["${BLINDTRANSFER}${ATTENDEDTRANSFER}" !=
""]?Set(REC_STATUS=NO))
exten => s,n(next),GotoIf($[${LEN(${ARG1})}]?checkaction)
exten => s,n(recorderror),Playback(something-terribly-wrong,error)
exten => s,n,Hangup
exten => s,n(checkaction),GotoIf($[${DIALPLAN_EXISTS(sub-record-
check,${ARG1})}]?sub-record-check,${ARG1},1)
exten => s,n,Noop(Generic ${ARG1} Recording Check - ${FROMEXTEN}
${ARG2})
exten => s,n,Gosub(recordcheck,1(${ARG3},${ARG1},${ARG2}))
exten => s,n,Return()

exten => recordcheck,1,Noop(Starting recording check against ${ARG1})
exten => recordcheck,n,Goto(${ARG1})
exten => recordcheck,n(dontcare),Return()
exten => recordcheck,n(always),Noop(Detected legacy "always" entry.
Mapping to "force")
exten => recordcheck,n(force),Set(__REC_POLICY_MODE=FORCE)
exten => recordcheck,n,GotoIf($["${REC_STATUS}"!="RECORDING"]?startrec)
exten => recordcheck,n,Return()
exten => recordcheck,n(delayed),Noop(Detected legacy "delayed" entry.
Mapping to "yes")
exten => recordcheck,n(yes),ExecIf($["${REC_POLICY_MODE}" = "NEVER" |
"${REC_POLICY_MODE}" = "NO" | "${REC_STATUS}" = "RECORDING"]?Return())
exten => recordcheck,n,Set(__REC_POLICY_MODE=YES)
exten => recordcheck,n,Goto(startrec)
exten => recordcheck,n(no),Set(__REC_POLICY_MODE=NO)
exten => recordcheck,n,Return()
exten => recordcheck,n(never),Set(__REC_POLICY_MODE=NEVER)
exten => recordcheck,n,Goto(stoprec)
exten => recordcheck,n(startrec),Noop(Starting recording: ${ARG2},
${ARG3})
exten => recordcheck,n,Set(AUDIOHOOK_INHERIT(MixMonitor)=yes)
exten =>
recordcheck,n,ExecIF($[${LEN(${FETGUZREALDST})}]?NoOP():Set(CDR(realdst
)=${ARG3}))
exten =>
recordcheck,n,ExecIF($[${LEN(${FETGUZREALSRC})}]?NoOP():Set(CDR(realsrc
)=${FROMEXTEN}))
exten => recordcheck,n,Set(__CALLFILENAME=${ARG2}-${ARG3}-${FROMEXTEN}-
${TIMESTR}-${UNIQUEID})
exten => recordcheck,n,Gosub(sub-check_recordfiles,s,1(${ARG2}))
exten =>
recordcheck,n,MixMonitor(${MIXMON_DIR}${YEAR}/${MONTH}/${DAY}/${CALLFIL
ENAME}.${MON_FMT},abi(LOCAL_MIXMON_ID)${MIXMON_BEEP},${MIXMON_POST})
exten => recordcheck,n,Set(__MIXMON_ID=${LOCAL_MIXMON_ID})
```

```
exten => recordcheck,n,Set(__RECORD_ID=${CHANNEL(name)})
exten => recordcheck,n,Set(__REC_STATUS=RECORDING)
exten =>
recordcheck,n,Set(CDR(recordingfile)=${CALLFILENAME}.${MON_FMT})
exten => recordcheck,n,Return()
exten => recordcheck,n(stoprec),Noop(Stopping recording: ${ARG2},
${ARG3})
exten => recordcheck,n,Set(__REC_STATUS=STOPPED)
exten => recordcheck,n,System(${AMPBIN}/stoprecording.php
"${CHANNEL(name)}")
exten => recordcheck,n,Return()

exten => out,1,Noop(Outbound Recording Check from ${FROMEXTEN} to
${ARG2})
exten => out,n,Set(CDR(realsrc)=${FROMEXTEN})
exten => out,n,Set(CDR(realdst)=${ARG2})
exten => out,n,Set(CUTTEDCHANNEL=${CUT(CHANNEL,,1)})
exten => out,n,Set(CUTTEDCHANNEL=${CUT(CUTTEDCHANNEL,/,2)})
exten => out,n,ExecIf($["${FROMEXTEN}" =
"${ARG2}"]?Set(CDR(realsrc)=${CUTTEDCHANNEL}))
exten =>
out,n,Set(RECMODE=${DB(AMPUSER/${FROMEXTEN}/recording/out/external)})
exten => out,n,ExecIf($[!${LEN(${RECMODE})} | "${RECMODE}" =
"dontcare"]?Goto(routewins))
exten => out,n,ExecIf($["${ARG3}" = "never" | "${ARG3}" =
"force"]?Goto(routewins))
exten => out,n(extenwins),Gosub(recordcheck,1(${RECMODE},out,${ARG2}))
exten => out,n,Return()
exten => out,n(routewins),Gosub(recordcheck,1(${ARG3},out,${ARG2}))
exten => out,n,Return()

exten => in,1,Noop(Inbound Recording Check to ${ARG2})
exten => in,n,Set(FROMEXTEN=unknown)
exten =>
in,n,ExecIf($[${LEN(${CALLERID(num)})}]?Set(FROMEXTEN=${CALLERID(num)})
)
exten => in,n,Set(_FETGUZREALDST=${ARG2})
exten => in,n,Set(_FETGUZREALSRC=${FROMEXTEN})
exten => in,n,Set(CDR(realsrc)=${FETGUZREALSRC})
exten => in,n,Set(CDR(realdst)=${FETGUZREALDST})
exten => in,n,Gosub(recordcheck,1(${ARG3},in,${ARG2}))
exten => in,n,Return()

exten => exten,1,Noop(Exten Recording Check between ${FROMEXTEN} and
${ARG2})
exten =>
exten,n,ExecIF($[${LEN(${FETGUZREALDST})}]?NoOP():Set(CDR(realdst)=${AR
G2}))
exten =>
exten,n,ExecIF($[${LEN(${FETGUZREALSRC})}]?NoOP():Set(CDR(realsrc)=${FR
OMEXTEN}))
```

```
exten => exten,n,Set(CDR(cnum)=${FROMEXTEN})
exten =>
exten,n,Set(CALLTYPE=${IF($[${LEN(${FROM_DID})}]?external:internal)})
exten =>
exten,n,ExecIf(${LEN(${CALLTYPE_OVERRIDE})}?Set(CALLTYPE=${CALLTYPE_OVE
RRIDE}))
exten =>
exten,n,Set(CALLEE=${DB(AMPUSER/${ARG2}/recording/in/${CALLTYPE})})
exten => exten,n,ExecIf($[!${LEN(${CALLEE})}]?Set(CALLEE=dontcare))
exten => exten,n,GotoIf($["${CALLTYPE}"="external"]?callee)
exten => exten,n,GotoIf($["${CALLEE}"="dontcare"]?caller)
exten =>
exten,n,ExecIf($[${LEN(${DB(AMPUSER/${FROMEXTEN}/recording/priority)})}
]?Set(CALLER_PRI=${DB(AMPUSER/${FROMEXTEN}/recording/priority)}):Set(CA
LLER_PRI=0))
exten =>
exten,n,ExecIf($[${LEN(${DB(AMPUSER/${ARG2}/recording/priority)})}]?Set
(CALLEE_PRI=${DB(AMPUSER/${ARG2}/recording/priority)}):Set(CALLEE_PRI=0
))
exten =>
exten,n,GotoIf($["${CALLER_PRI}"="${CALLEE_PRI}"]?${REC_POLICY}:${IF($[
${CALLER_PRI}>${CALLEE_PRI}]?caller:callee)})
exten =>
exten,n(callee),Gosub(recordcheck,1(${CALLEE},${CALLTYPE},${ARG2}))
exten => exten,n,Return()
exten =>
exten,n(caller),Set(RECMODE=${DB(AMPUSER/${FROMEXTEN}/recording/out/int
ernal)})
exten => exten,n,ExecIf($[!${LEN(${RECMODE})}]?Set(RECMODE=dontcare))
exten =>
exten,n,ExecIf($["${RECMODE}"="dontcare"]?Set(RECMODE=${CALLEE}))
exten => exten,n,Gosub(recordcheck,1(${RECMODE},${CALLTYPE},${ARG2}))
exten => exten,n,Return()

exten => conf,1,Noop(Conference Recording Check ${FROMEXTEN} to
${ARG2})
exten => conf,n,Gosub(recconf,1(${ARG2},${ARG2},${ARG3}))
exten => conf,n,Return()

exten => page,1,Noop(Paging Recording Check ${FROMEXTEN} to ${ARG2})
exten =>
page,n,GosubIf($["${REC_POLICY_MODE}"="always"]?recconf,1(${ARG2},${FRO
MEXTEN},${ARG3}))
exten => page,n,Return()

exten => recconf,1,Noop(Setting up recording: ${ARG1}, ${ARG2},
${ARG3})
exten =>
recconf,n,Set(__CALLFILENAME=${IF($[${CONFBRIDGE_INFO(parties,${ARG2})}
]?${DB(RECCONF/${ARG2})}:${ARG1}-${ARG2}-${ARG3}-${TIMESTR}-
${UNIQUEID})})
```

```
exten =>
recconf,n,ExecIf($[!${CONFBRIDGE_INFO(parties,${ARG2})}]?Set(DB(RECCONF
/${ARG2})=${CALLFILENAME}))
exten =>
recconf,n,Set(CONFBRIDGE(bridge,record_file)=${MIXMON_DIR}${YEAR}/${MON
TH}/${DAY}/${CALLFILENAME}.${MON_FMT})
exten => recconf,n,ExecIf($["${ARG3}"!="always"]?Return())
exten => recconf,n,Set(CONFBRIDGE(bridge,record_conference)=yes)
exten => recconf,n,Set(__REC_STATUS=RECORDING)
exten =>
recconf,n,Set(CDR(recordingfile)=${IF($[${CONFBRIDGE_INFO(parties,${ARG
2})}]?${CALLFILENAME}.${MON_FMT}:${CALLFILENAME}.${MON_FMT})})
exten => recconf,n,Noop(${MIXMONITOR_FILENAME})
exten => recconf,n,Set(CHANNEL(hangup_handler_push)=sub-record-hh-
check,s,1)
exten => recconf,n,Return()

exten => recq,1,Noop(Setting up recording: ${ARG1}, ${ARG2}, ${ARG3})
exten => recq,n,Set(AUDIOHOOK_INHERIT(MixMonitor)=yes)
exten =>
recq,n,Set(MONITOR_FILENAME=${MIXMON_DIR}${YEAR}/${MONTH}/${DAY}/${CALL
FILENAME})
exten =>
recq,n,MixMonitor(${MONITOR_FILENAME}.${MON_FMT},${MONITOR_OPTIONS}${MI
XMON_BEEP},${MIXMON_POST})
exten => recq,n,Set(__REC_STATUS=RECORDING)
exten => recq,n,Set(CDR(recordingfile)=${CALLFILENAME}.${MON_FMT})
exten => recq,n,Return()

exten => parking,1,Noop(User ${ARG2} picked up a parked call)
exten => parking,n,Set(USER=${ARG2})
exten => parking,n,ExecIf($[!${LEN(${ARG2})}]?Set(USER=unknown))
exten =>
parking,n,Set(RECMODE=${DB(AMPUSER/${ARG2}/recording/out/internal)})
exten => parking,n,ExecIf($[!${LEN(${RECMODE})}]?Set(RECMODE=dontcare))
exten => parking,n,Gosub(recordcheck,1(${RECMODE},parked,${USER}))
exten => parking,n,Return()
```

после чего выполните команду

```
dialplan reload
```

в консоли Asterisk
Код проверен на FreePBX 13.0.191.11 и 14.0.2.10
Если мажорная (13 в данном примере) версия FreePBX у вас отличается, то контекст [sub-record-check] может быть немного другим. Напишите нам на support@fetg.uz и мы поможем с написанием диалплана данного контекста.

6. С cdr разобрались, дело за queue_log. Эта таблица нужна нам для того, чтобы понимать какие события происходили в очередях Asterisk. Перво-наперво создадим ее

```
CREATE TABLE `queue_log` (
 `id` INT(10) UNSIGNED NOT NULL AUTO_INCREMENT,
 `time` TIMESTAMP NULL DEFAULT '0000-00-00 00:00:00',
 `callid` VARCHAR(32) NOT NULL DEFAULT '',
 `queuename` VARCHAR(32) NOT NULL DEFAULT '',
 `agent` VARCHAR(32) NOT NULL DEFAULT '',
 `event` VARCHAR(32) NOT NULL DEFAULT '',
 `data1` VARCHAR(100) NOT NULL DEFAULT '',
 `data2` VARCHAR(100) NOT NULL DEFAULT '',
 `data3` VARCHAR(100) NOT NULL DEFAULT '',
 `data4` VARCHAR(100) NOT NULL DEFAULT '',
 `data5` VARCHAR(100) NOT NULL DEFAULT '',
 PRIMARY KEY (`id`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8 COLLATE=utf8_unicode_ci;
```

7. Научим FreePBX(читай Asterisk) писать туда все то, что нам нужно.
   Отправляемся в web-интерфейс по знакомой дорожке и создаем новый файл под именем **extconfig.conf**, кликнув Add New File.
   Внести в него нужно лишь одну строчку

   ```
   [settings]
   queue_log => odbc,asteriskcdrdb
   ```

   Скрин: *не забываем про Save и Apply Configs
   После выполнения данных действий и совершения звонков в очереди, запрос из консоли mysql

   ```
   SELECT * FROM queue_log;
   ```

   должен вернуть вам нечто подобное *на запрос в скрине не ориентируйтесь, там я специально добавил условие, чтобы сократить выхлоп

8. С препарированием нашей АТС по части БД закончили!
   Переходим к настройке синхронизации БД и файлов записей разговоров в облако FETG.UZ. Для этого воспользуйтесь инструкцией - https://wiki.fetg.uz/doku.php?id=cloud_daemon_sync
9. Для входа в систему статистики используйте данные авторизации (Email-адрес/Пароль) из регистрационного письма.

На этом все настройки завершены.
Если вы все сделали правильно, то через какое-то время (зависит от размера БД) сможете воспользоваться всеми отчетами системы статистики FETG.UZ в своем личном кабинете.
Повторюсь, если у вас остались вопросы и/или есть предложения/замечания, пишите нам на support@fetg.uz.