Kamailio как PUSH Mid-Registrar

KAMAILIO CONF'24

Обо мне:

- Компания IQTEK
- Разработка и внедрение на базе OSS VoIP
- Начинал с Asterisk 0.99
- Asterisk DCAP/DCAA

О чем доклад

- Для чего нужен Push
- Вариант задачи
- Понятие AoR throttling
- Скринживаем 🐍 и 🦔

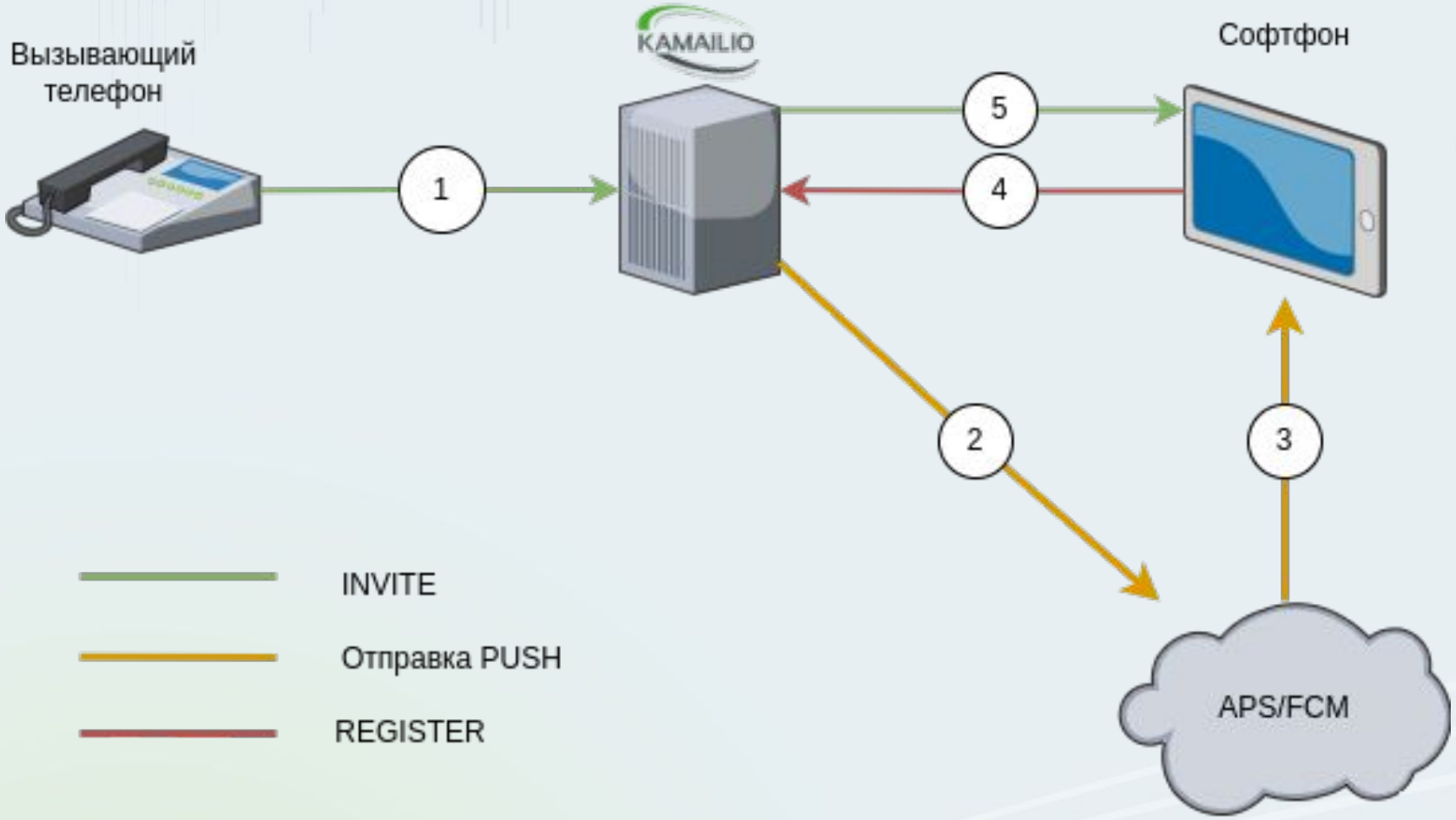
Работа мобильного приложения

- потребление энергии
- потребление трафика
- ненадежность сокета
- остановка приложения мобильной ОС

Push

- экономия ресурсов
- использование приложения по запросу
- отсутствие необходимости непрерывно обновлять AOR
- нет необходимости в поддержании NAT

Работа мобильного приложения



Задача

- Мобильное приложение
- Обеспечить надежные входящие вызовы
- подключении к произвольному SIP серверу
- Fork вызова на несколько устройств
- На АТС состояние регистрации должно быть корректно

“Даже” в Asterisk представляет проблему:

- Нет способа отследить получение регистрации
- Необходимо скрещивать Originate и Confbridge
- Альтернатива - ARI

REGISTER: от АТС на софтфон (v1 - uac)

- Kamailio получает запрос на регистрацию от АТС
- Регистрируем клиента
- Включаем uac_rec для данного AOR
- При завершении - отключаем

НО!

- UAC в отдельном потоке и по таймеру
- Включение и отключение регистрации с задержкой

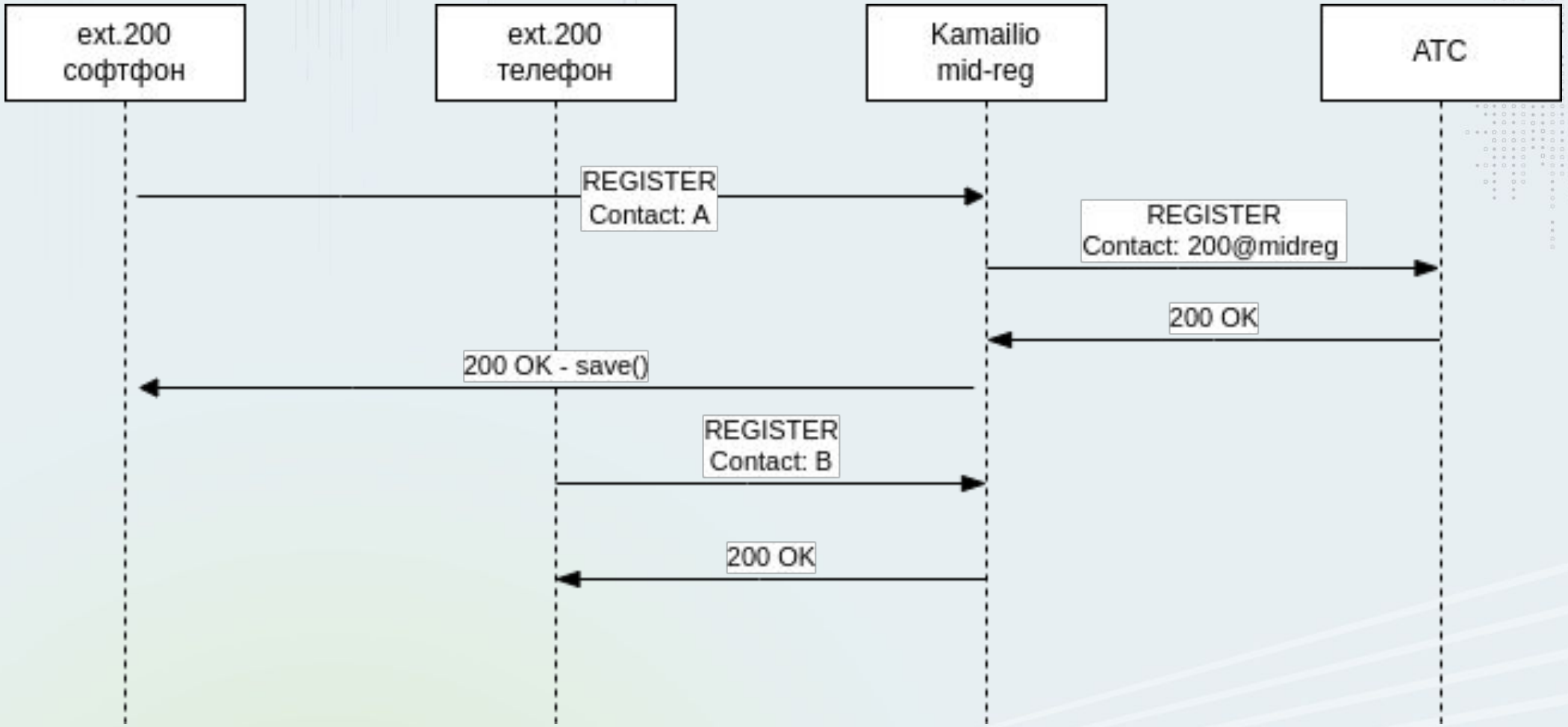
REGISTER: от АТС на софтфон

- Kamailio получает запрос на регистрацию от АТС
- Заменяем Contact
- Для корректной авторизации - пропускаем все новые Register на АТС
- Вызываем save() только на получение 200 ок от АТС
- На основе User-Agent сохраняем тип устройства

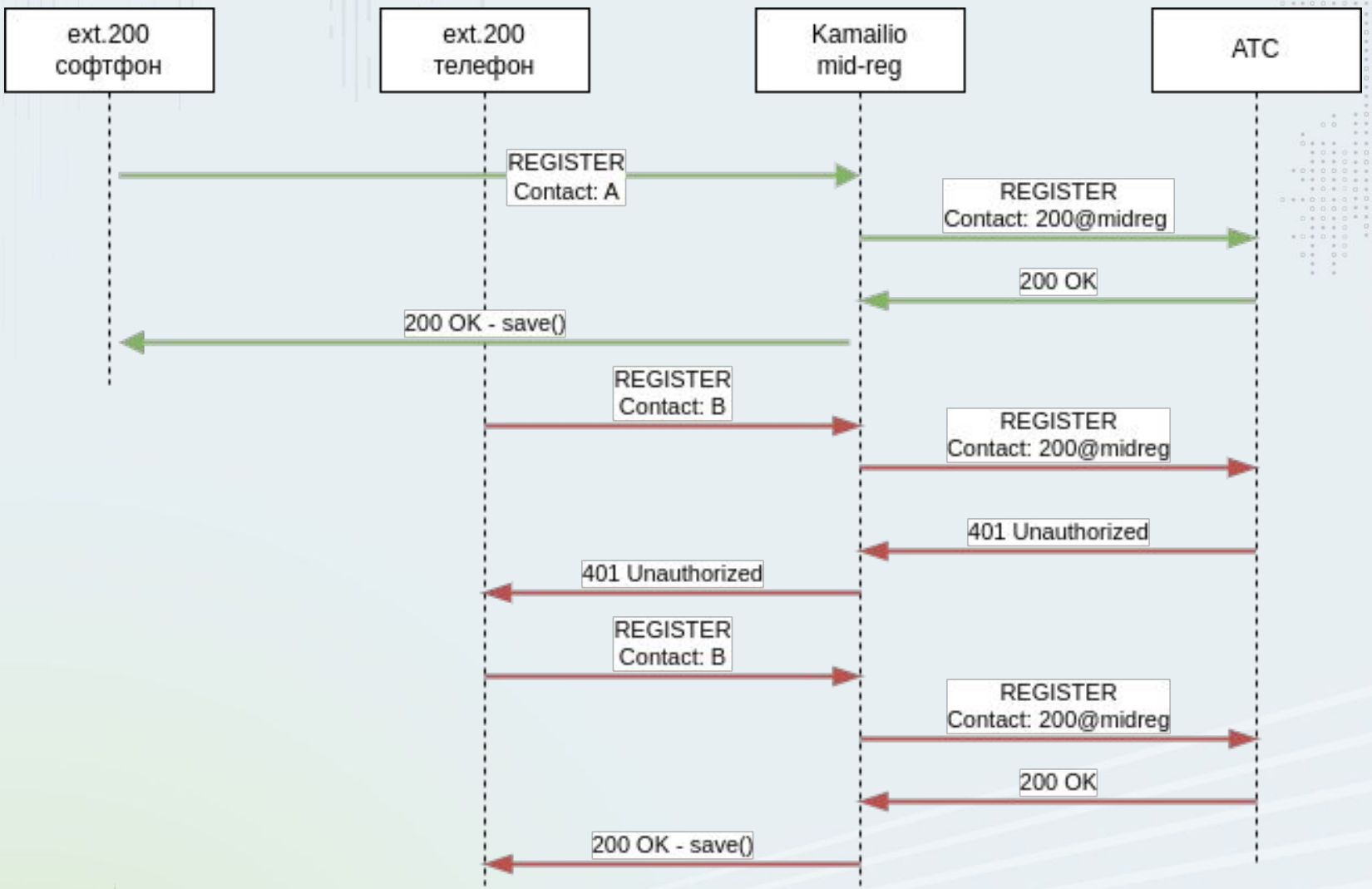
В случае истечения срока регистрации:

- Проверяем активность токена
- Отправляем самостоятельно запрос на регистрацию (send_data)

REGISTER: AOR-throttling (opensips)



REGISTER: AOR-throttling



INVITE: от АТС на софтфон

- АТС отправляет SIP INVITE на Kamailio
- Kamailio обнаруживает все контакты и тип подключенного устройства
- Для мобильного устройства - отправляется PUSH
  - Если PUSH вернул флаг отсутствия активного токена - снимаем флаг типа
- Для активных устройств INVITE отправляется незамедлительно
- При получении REGISTER от мобильного клиента - присоединяем к транзакции

tsilo

- хранение транзакций до момента реакции устройства
- http_async_client - обращение к push сервису (абстракцией над APNS/Apple или FCM/Google)
- usrloc интегрирован с tsilo и может добавлять branch транзакции по мере регистрации

Зависит от:

- usrloc
- tm / tmx
- registrar

tsilo

- **ts_store([uri])**
  - Сохранение текущей транзакции в хэше tsilo
- **ts_append(domain, ruri)**
  - добавить бранчи ко всем транзакциям по URI

Дополнительные функции (tmx):

- **t_suspend()**
  - Приостановка транзакции - для случая отсутствия активных устройств
- **t_continue()**
  - Возобновление транзакции - для первой регистрации

ts_append(...)

- При получении REGISTER
- Добавляет branch во все активные транзакции по заданному URI

```
lock("$tU");
$var(hjoin) = $sht(vtp=>join::$tU);
$var(hstored) = $sht(vtp=>stored::$tU);
xlog("resuming $tU $var(hjoin) $var(hstored)\n");
$sht(vtp=>join::$tU) = $null;
unlock("$tU");
# Добавляем branch к активной транзакции
if ($var(hjoin)==0) {
  if ($var(hstored)) {
    xdbg("APPEND: $tU $avp(RECEIVED) $avp(ct) $ct\n");
    ts_append("location", "$(ct{s.unbracket})");
  }
  return;
}
```

t_suspend()

- При получении входящего INVITE
- “Замораживает” транзакцию
- Далее отправляется запрос на PUSH сервер

```
# используется только в случае если не было ни одного location!!!
route[SUSPEND] {
 if(!t_suspend())
 {
  xlog("failed suspending trasaction [$T(id_index): $T(id_label)]\n");
  send_reply("501", "Unknown destination");
  exit;
 }

 xdbg("suspended transaction [$T(id_index):$T(id_label)] $fU => $rU\n");
  $sht(vtp=>id_index::$rU) = $T(id_index);
  $sht(vtp=>id_label::$rU) = $T(id_label);
  $sht(vtp=>join::$rU) = "" + $T(id_index) + ":" + $T(id_label);
 xdbg("htable key value [$sht(vtp=>join::$rU)]\n");
}
```

t_continue(...)

- При получении REGISTER
- Возобновляет указанную транзакцию

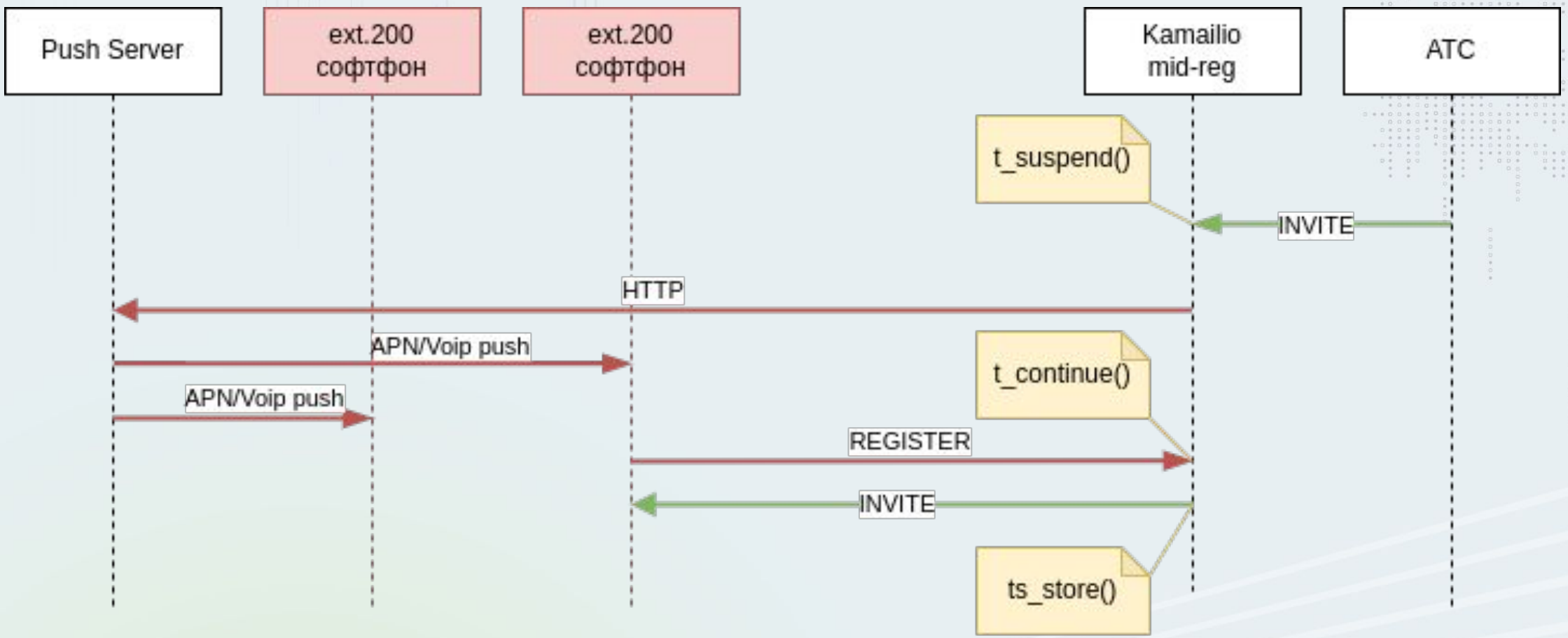
```
# Нет активной транзакции - возобновляем остановленную через t_suspend()
$var(id_index) = $(var(hjoin){s.select,0,:}{s.int});
$var(id_label) = $(var(hjoin){s.select,1,:}{s.int});
xlog("resuming trasaction [$var(id_index):$var(id_label)] $tU ($var(hjoin))\n");
t_continue("$var(id_index)", "$var(id_label)", "INVRESUME");
```

t_store(...)

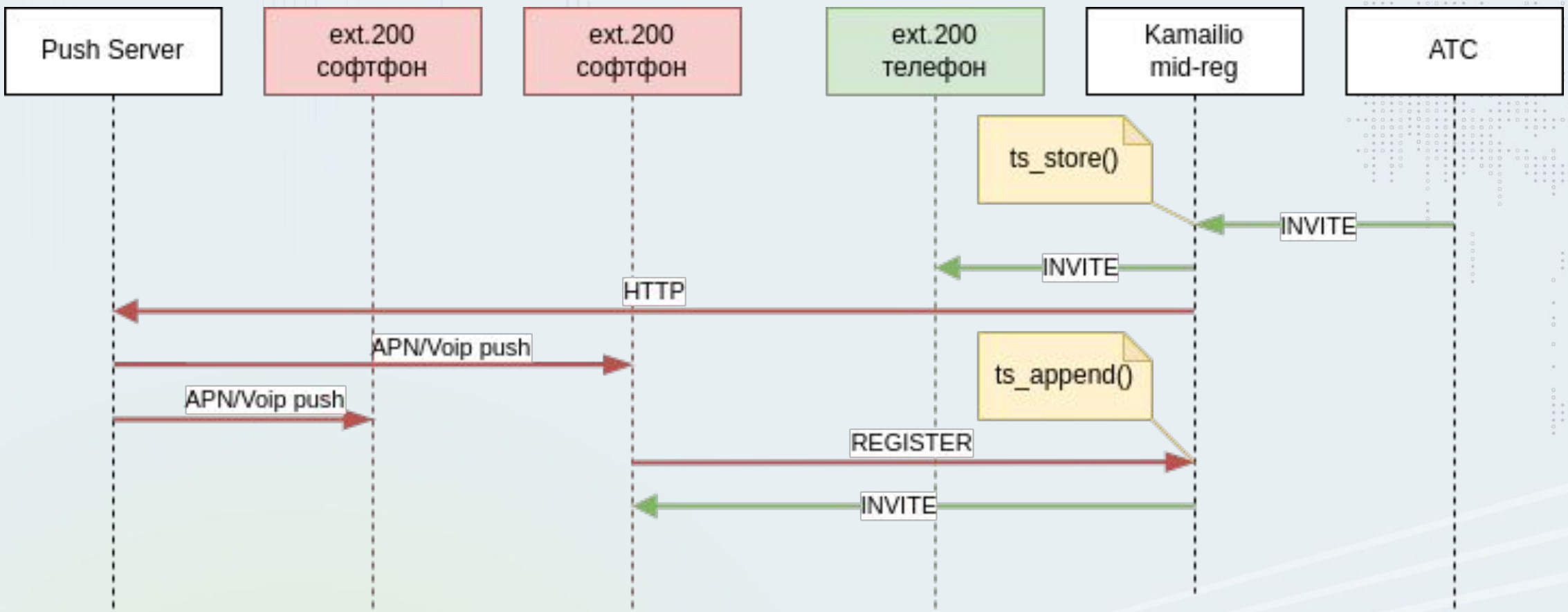
- При получении транзакции INVITE
  - или после вызова t_continue
- Сохранение данных транзакции в хэш таблице tsilo

```
# lookup and relay after resuming transaction
route[INVRESUME] {
  lookup("location");
  t_relay();
  ts_store("sip:$rU@MY_PUBLIC_IP:5060");
  $sht(vtp=>stored::$rU) = 1;
  xdbg("stored transaction [$T(id_index):$T(id_label)] $fU => $rU\n");
}
```

tsilo


Push Server
ext.200
софтфон
ext.200
софтфон
Kamailio
mid-reg
ATC
t_suspend()
INVITE
HTTP
APN/Voip push
t_continue()
APN/Voip push
REGISTER
INVITE
ts_store()

tsilo


Push Server
ext.200 софтфон
ext.200 софтфон
ext.200 телефон
Kamailio mid-reg
ATC
ts_store()
INVITE
INVITE
HTTP
APN/Voip push
ts_append()
APN/Voip push
REGISTER
INVITE

Заключение

- Kamailio предоставляет инструментарий для реализации гибких сценариев
- Не всё получается с первого раза
- Не всё получается со второго раза
- Не освещены вопросы:
  - Обработка 183 ответов от нескольких вопросов
  - Реализации PUSH сервера
  - Различные варианты поведения при разрыве вызова

Спасибо за внимание!

Буду рад ответить на все ваши вопросы
сейчас или свяжитесь со мной в будущем:

IQTEK

Гончаровский Игорь

igorg@iqtek.ru

@IgorrG