GS LABS
GS GROUP

Публичный - L0

# Сервер передачи ключей Keys Transfer Server (KTS)

## Руководство администратора

| | |
|---|---|
| Индекс | KTS-AG |
| Конфиденциальность | Публичный - L0 |
| Ревизия | 1.0 |
| Статус | Согласован |

# Содержание

# 1. Аннотация

Данный документ содержит настройки компонентов Сервера передачи ключей Keys Transfer Server (KTS) (далее по тексту - KTS или Система).

Документ предназначен для сотрудников отдела мониторинга и инсталляции, а также для других технических специалистов, в обязанности которых входит настройка системы KTS и поддержание её работоспособности.

## 2. Термины и сокращения

| Термин | Определение |
|---|---|
| FTP (File Transfer Protocol) | Протокол передачи файлов – стандартный протокол, предназначенный для передачи файлов по TCP-сетям (например, Интернет). Протокол построен на архитектуре "клиент-сервер" и использует разные сетевые соединения для передачи команд и данных между клиентом и сервером. |
| Fusemap Config | Подробное описание области OTP-памяти чипа с адресами, полями и их размерами, включая OTP-ключи и конфигурационные биты. |
| KGS | Система генерации ключей Keys Generation System (KGS). Продукт ООО "ПЦТ" для работы с ключами: генерация, экспорт, импорт, управление. |
| KTS сервер | Сервер, на котором развернуты bbx_server_go и база данных (компоненты "Сервера передачи ключей Keys Transfer Server (KTS)"). |
| OTP-ключи (One Time Programmable) | Ключи, которые прошиваются в однократно программируемую область памяти в чипе. |
| Номер Партии (Part Number, PN) | Сущность, которая характеризует заказы чипов у производителя (чип-вендора). Как правило, в чипах используемое имя part number, выгравированное на корпусе. В KGS сущность кроме имени имеет внутренний идентификатор.<br><br>В KGS имя задается пользователем и может не совпадать с маркировкой на чипе. |
| Тип Партии (Part Type, PT, PTTP) | Сущность, которая характеризует набор общих OTP-ключей в чипах одной партии: у всех чипов одной PT одинаковые общие ключи. В чипах используется только идентификатор Part Type ID, в KGS также используется пользовательское имя.<br><br>Не следует путать Тип Партии с Номером Партии. |

| Сокращение | Расшифровка |
|---|---|
| БД | База данных |
| FMC | FuseMap Config |
| FMCD | FuseMap Config Decription |
| ID | Identifier |
| OTP | One-Time Programmable |
| TDE | Transparent Database Encryption |

# 3. Обслуживание системы

Обслуживание системы KTS заключается в выполнении следующих основных действий:

- Изменение настроек KTS (при необходимости).
- Эксплуатация системы: управление данными в базе KTS с помощью специальных скриптов.
- Устранение ошибок на основе логов системы.

# 4. Настройка bbx_server_go

Настройка осуществляется с помощью конфигурационного файла *bbx_server_go.cfg*.

## 4.1. Конфигурационный файл bbx_server_go

Изначально в системе присутствует шаблон файла конфигурации с расширением ***.cfg.dft***, содержащий дефолтные значения параметров.

Создайте копию этого файла с расширением *.cfg*, с которой в дальнейшем будет производиться работа:

```
sudo cp bbx_server_go.cfg.dft bbx_server_go.cfg
```

Таким образом, шаблон с начальными  значениями параметров остается неизменным.

Содержание файла конфигурации приведено в таблице:

| Параметр | Описание |
|---|---|
| LOGGER_LEVEL | Степень логирования событий.<br><br>Возможные значения:<br><br>• trace,<br>• debug,<br>• info (значение по умолчанию),<br>• warning,<br>• error,<br>• fatal.<br><br>Для тестирования рекомендуется "debug" или "trace".<br><br>Подробное описание уровней логирования приведено в разделе ниже. |
| LOGGER_OUTPUT_FILE | Параметр для вывода логов в файл. В параметре указывается полный путь до файла, куда пишутся логи.<br><br>Если параметр пустой или отсутствует, то логи пишутся в stdout, если параметр задан, то логи пишутся в файл. |

| | |
|---|---|
| SYSTEM_TRACER_ENABLED | Настройки взаимодействия с системой отслеживания запросов.<br><br>Флаг включения трассировки. Возможные значения:<br><br>• false - трассировка выключена.<br>• true - трассировка включена. |
| SYSTEM_TRACER_AGENT_HOST_PORT | Адрес и номер порта agent, т.е. сервера, на который идет отправка параметров трассировки. |
| SYSTEM_HTTP_ADDRESS | Адрес для запуска сервера в контейнере. Формат записи: *<IP-адрес>:<номер порта>*<br><br>Параметр является общим (т.е. адрес и порт - общие) для 2 протоколов: http и https |
| SYSTEM_HTTP_WRITE_TIMEOUT | Таймаут (в сек.), по которому будет разорвано соединение (при обращении на запись данных). |
| SYSTEM_HTTP_READ_TIMEOUT | Таймаут (в сек.), по которому будет разорвано соединение (при обращении на чтение данных). |
| SYSTEM_HTTP_IS_USED | Флаг включения взаимодействия по http.<br><br>Возможные значения:<br><br>• false - Взаимодействие выключено.<br>• true - Сервис взаимодействует по http. |
| SYSTEM_HTTPS_ENABLED | Флаг включения взаимодействия по https, т.е. используется или нет протокол **защищенной** сети ( = включение работы с SSL сертификатами). Возможные значения:<br><br>• true - секция включена;<br>• false - секция выключена.<br><br>Если параметр отсутствует, то считается, что = false. |

| | |
|---|---|
| SYSTEM_HTTPS_VERIFY_MODE | Режим работы с сертификатами.<br><br>Возможные значения:<br><br>• 0 - NoClientCert ClientAuthType = iota<br>NoClientCert означает, что сертификат клиента не будет запрашиваться во время взаимодействия. Если какие-либо сертификаты будут отправлены, то они не будут подвергаться проверке.<br>• 1 - RequestClientCert<br>RequestClientCert означает, что сертификат клиента должен быть запрошен во время взаимодействия, однако не требует, чтобы клиент отправлял какие-либо сертификаты.<br>• 2 - RequireAnyClientCert<br>RequireAnyClientCert означает, что сертификат клиента должен быть запрошен во время взаимодействия, и что по крайней мере один сертификат должен быть отправлен клиентом, но этот сертификат не обязательно должен быть действительным (валидным).<br>• 3 - VerifyClientCertIfGiven<br>VerifyClientCertIfGiven означает, что сертификат клиента должен быть запрошен во время взаимодействия, но не требует, чтобы клиент отправлял сертификат. Если клиент отправляет сертификат, он должен быть действительным.<br>• 4 - RequireAndVerifyClientCert<br>RequireAndVerifyClientCert означает, что сертификат клиента должен быть запрошен во время взаимодействия, и что клиент должен отправить по крайней мере один действительный сертификат.<br><br>Если параметр отсутствует, то считается, что = 0. |
| SYSTEM_HTTPS_CA_CERTIFICATE | Путь к файлу с CA сертификатом. Обязательное поле. |
| SYSTEM_HTTPS_SERVER_CERTIFICATE | Путь к файлу с server certificate. Обязательное поле. |
| SYSTEM_HTTPS_SERVER_KEY | Путь к файлу с server key. Обязательное поле. |
| SYSTEM_DB_MASTER_HOST | IP-адрес мастер реплики (инстанс кластера KTS DB) |
| SYSTEM_DB_MASTER_PORT | Номер порта мастер реплики (инстанс кластера KTS DB) |
| SYSTEM_DB_MASTER_USER | Логин пользователя, под которым осуществляется подключение к мастер реплике (инстансу кластера KTS DB) |
| SYSTEM_DB_MASTER_PASSWORD | Пароль для подключения к мастер реплике (инстансу кластера KTS DB) |
| SYSTEM_DB_MASTER_DB_NAME | Имя мастер реплики (инстанс кластера KTS DB) в PostgreSQL |

| | |
|---|---|
| SYSTEM_DB_MASTER_TIMEOUT | Время ожидания ответа (в сек.) от мастер реплики (инстанс кластера KTS DB) |
| SYSTEM_DB_MASTER_MAX_CONNS | Максимальное количество соединений, создаваемых сервером с мастер репликой (инстансом кластера KTS DB) |
| SYSTEM_DB_ASYNC_REPLICA_HOST | IP-адрес асинхронной реплики (инстанс кластера KTS DB) |
| SYSTEM_DB_ASYNC_REPLICA_PORT | Номер порта асинхронной реплики (инстанс кластера KTS DB) |
| SYSTEM_DB_ASYNC_REPLICA_USER | Логин пользователя, под которым осуществляется подключение к асинхронной реплике (инстансу кластера KTS DB) |
| SYSTEM_DB_ASYNC_REPLICA_PASSWORD | Пароль для подключения к асинхронной реплике (инстансу кластера KTS DB) |
| SYSTEM_DB_ASYNC_REPLICA_DB_NAME | Имя асинхронной реплики (инстанс кластера KTS DB) в PostgreSQL. |
| SYSTEM_DB_ASYNC_REPLICA_TIMEOUT | Время ожидания ответа (в сек.) от асинхронной реплики (инстанс кластера KTS DB) |
| SYSTEM_DB_ASYNC_REPLICA_MAX_CONNS | Максимальное количество соединений, создаваемых сервером с асинхронной репликой (инстансом кластера KTS DB) |
| SYSTEM_PROMETHEUS_HTTP_ENABLED | Флаг включения прослушки Prometheus.<br><br>Возможные значения:<br><br>• true - секция включена;<br>• false - секция выключена. |
| SYSTEM_PROMETHEUS_HTTP_ADDRESS | Адрес для запуска сервера в контейнере. Формат записи: *<IP-адрес>:<номер порта>* |
| SYSTEM_PROMETHEUS_HTTP_WRITE_TIMEOUT | Таймаут (в сек.), по которому будет разорвано соединение (при обращении на запись данных). |
| SYSTEM_PROMETHEUS_HTTP_READ_TIMEOUT | Таймаут (в сек.), по которому будет разорвано соединение (при обращении на чтение данных). |
| SYSTEM_PPROF_ENABLED | Флаг включения взаимодействия с pprof. Возможные значения:<br><br>• false - pprof выключен. Значение по умолчанию.<br>• true - pprof включен.<br><br>**Примечание**. Подробное описание приведено в разделе ниже. |

| | |
|---|---|
| SYSTEM_PPROF_ACCESS_KEY | Значение ключа доступа, отправляемого в запросе к pprof. Запрос используется для профилирования сервисов (исследования CPU). |
| SYSTEM_TDE_DSN | DSN базы данных (KTS DB), которую использует tde (механизм шифрования данных по лестнице ключей) |
| SYSTEM_TDE_HWRK_FILE_NAME | Полный путь и название private HWRK-ключа, сгенерированного на сервере KTS<br><br>**Примечание**. Ключ является частью лестницы ключей, применяемой при шифровании данных в KTS DB |
| SYSTEM_TDE_DBMK_FILE_NAME | Полный путь и название BBMK-ключа, шифрованного public HWRK-ключом. Файл с ключом экспортируется из системы KGS<br><br>**Примечание**. Ключ является частью лестницы ключей, применяемой при шифровании данных в KTS DB |
| SYSTEM_TDE_USE_MLOCK | Флаг. Если установлен в true, то блокируется возможность сброса на диск данных, хранящихся в оперативной памяти в модуле по работе с tde |

Пример файла:

**bbx_server_go.cfg**

```
{
  "logger": {
      "level": "debug",
      "output_file": ""
  },
  "system" : {
    "tracer" : {
        "enabled" : false,
        "agent_host_port": "agent:6831"
    },
    "http": {
      "address": "0.0.0.0:8080",
      "write_timeout": 45,
      "read_timeout": 15,
      "is_used" : true
    },
    "https": {
      "enabled": true,
      "verify_mode": 2,
      "ca_certificate": "cert/cacert.pem",
      "server_certificate": "cert/server-cert.pem",
      "server_key": "cert/server-key.pem"
    },
    "db" : {
        "master": {
           "host": "192.168.14.47",
           "port": 5432,
           "user": "bbxadmin",
           "password": "bbxadmin",
           "db_name": "bbx_server",
           "timeout" : 50,
           "max_conns" : 10
        },
        "async_replica": {
           "host": "192.168.14.47",
           "port": 5432,
           "user": "bbxadmin",
           "password": "bbxadmin",
           "db_name": "bbx_server",
           "timeout" : 50,
           "max_conns" : 10
         }
    },
    "prometheus_http" :
    {
       "address" : "0.0.0.0:9102",
       "write_timeout" : 5,
       "read_timeout" : 5
    },
    "pprof": {
        "enable" : true,
        "access_key" : "fc64a74a-51ca-4cb9-afea-5d5c633bc4d0"
    },
    "tde": {
        "dsn": "BBX_STAND",
        "hwrk_file_name": "/opt/chipblackbox/kmi_file11.dat",
        "dbmk_file_name": "/opt/chipblackbox/bbmk.dat",
        "use_mlock": true
    }
  }
}
```

## 4.2. Взаимодействие с pprof

Для анализа узких мест в реализации серверов и точечной настройки Golang серверов используется сторонний компонент pprof.

Для работы с ним в конфигурационном файле каждого Golang сервера, а также в default.yaml/production.yaml имеется секция pprof.

По умолчанию pprof выключен. Чтобы включить взаимодействие с pprof, необходимо:

- в production.yaml выставить pprof.enable равным true и заменить access_key на собственное значение:

```
...
pprof:
  enabled: true
  access_key: fc64a74a-51ca-4cb9-afea-5d5c633bc4d0
...
```

> ⚠ Значение pprof в production.yaml имеет более высокий приоритет, чем в конфигурационных файлах компонентов, поэтому менять значения в конфигурационном файле каждого отдельного сервиса не требуется.

- развернуть либо перезапустить все службы KTS.

Если pprof включен, то он позволяет отправлять запрос вида:

**Пример запроса**

```
curl --request GET '192.168.11.86:30170/debug/pprof/profile?seconds=15' --header 'X-Api-Key: fc64a74a-51ca-4cb9-afea-5d5c633bc4d0' --output fasentry.bin
```

Наличие параметра *--header 'X-Api-Key: значение_access_key_из_конфига'* в запросе обязательно.

Запрос используется для профилирования сервисов (исследования CPU).

# 5. Эксплуатация Системы

## 5.1. Перенос данных

Перенос персонализационных данных (OTP-ключей), а также прочих данных с FTP-сервера KGS на KTS сервер и обратно осуществляется администратором KTS. Файлы при передаче дополнительно шифруются PGP-ключом администратора.

### 5.1.1. Загрузка первоначальной и обновленной конфигурации на KTS сервер

Загрузка конфигурации осуществляется аналогично загрузке лестницы ключей.

1. Текстовый файл конфигурации переносится с FTP-сервера KGS в рабочую папку БД KTS и расшифровывается PGP-ключом администратора.

   ⚠ Файл с обновленной конфигурацией должен помещаться в папку, доступную для bbx_server_go.

2. Запускается скрипт *blbx_load_config.sh* (входит в комплект поставки).
   Параметры запуска:

   a. -help - вызов справки.
   b. -input_file "<input file>" - путь к файлу с загружаемыми данными (файл с конфигурацией для part type либо файл с лестницей ключей). **Параметр задаётся "в кавычках"** (т.е. если задан путь к файлу, то в кавычках указывается весь путь, если задано только имя файла (лежит в той же папке, что и скрипт), то в кавычках указывается имя файла). Обязательный параметр. Файл выгружается из KGS. Доступ к форматам файлов предоставляется по запросу.
   c. -h <host> - имя хоста KTS DB. Значение по умолчанию (задается в файле скрипта) - "localhost".
   d. -p <port> - номер порта KTS DB. Значение по умолчанию (задается в файле скрипта) - "5432".
   e. -u <pg_user> - имя пользователя KTS DB. Значение по умолчанию (задается в файле скрипта) - "bbxadmin".
   f. -P <pg_user_password> - пароль для доступа к KTS DB. Значение по умолчанию (задается в файле скрипта) - "bbxadmin".
   g. -d <database> - имя базы KTS DB. Значение по умолчанию (задается в файле скрипта) - "bbx".

   Формат команды запуска скрипта с параметрами:

   ```
   bash blbx_load_config.sh [-help] -input_file "<input file>" [-h <host>] [-p <port>] [-u <pg_user>] [-P <pg_user_password>] [-d <database>]
   ```

3. Скрипт осуществляет загрузку конфигурации в целевые структуры БД.
4. По окончании работы скрипта проверяется его лог файл (blbx_load_config.log) на отсутствие ошибок.

### 5.1.2. Загрузка новых OTP-ключей на KTS сервер

1. Файл с данными переносится с FTP-сервера KGS в рабочую папку БД KTS и расшифровывается PGP-ключом администратора.
2. Запускается скрипт *blbx_load_keys.sh* (входит в комплект поставки).
   Параметры запуска:
   a. -help - вызов справки.

b. -key_file "<keys file>" - путь к файлу с OTP-ключами. Обязательный параметр. Файл выгружается из KGS. Доступ к формату файла предоставляется по запросу (поддерживаются обе версии формата, но формат более поздней версии является предпочтительным).
**Параметр задаётся "в кавычках"** (т.е. если задан путь к файлу, то в кавычках указывается весь путь, если задано только имя файла (лежит в той же папке, что и скрипт), то в кавычках указывается имя файла).
c. -part_num <part number> - имя part number (Номера Партии), для которого загружаются ключи. Обязательный параметр.
d. -tde_hash "<tde hash file>" - путь к файлу с хешом лестницы ключей. Обязательный параметр. Файл выгружается из KGS. Доступ к формату файла предоставляется по запросу.
**Параметр задаётся "в кавычках"** (т.е. если задан путь к файлу, то в кавычках указывается весь путь, если задано только имя файла (лежит в той же папке, что и скрипт), то в кавычках указывается имя файла).
e. -h <host> - имя хоста KTS DB. Значение по умолчанию (задается в файле скрипта) - "localhost".
f. -p <port> - номер порта KTS DB. Значение по умолчанию (задается в файле скрипта) - "5432".
g. -u <pg_user> - имя пользователя KTS DB. Значение по умолчанию (задается в файле скрипта) - "bbxadmin".
h. -P <pg_user_password> - пароль для доступа к KTS DB. Значение по умолчанию (задается в файле скрипта) - "bbxadmin".
i. -d <database> - имя базы KTS DB. Значение по умолчанию (задается в файле скрипта) - "bbx".

Формат команды запуска скрипта с параметрами:

```
bash blbx_load_keys.sh [-help] -key_file "<keys file>" -part_num <part number> -tde_hash "<tde hash file>" [-h <host>] [-p <port>] [-u <pg_user>] [-P <pg_user_password>] [-d <database>]
```

⚠ **Обратите внимание!** При загрузке ключей из "<keys file>", имеющего формат более поздней версии, допустимо не указывать параметр *-part_num <part number>*

3. Скрипт осуществляет загрузку OTP-ключей в целевые структуры БД.
4. По окончании работы скрипта проверяется его лог файл (blbx_load_keys.log) на отсутствие ошибок.

### 5.1.3. Формирование и выгрузка отчета о программировании

Для формирования отчета о программировании используется скрипт *blbx_make_report.sh* (входит в комплект поставки).

В качестве аргументов указываются следующие параметры:

1. -help - вызов справки.
2. -f "<output file>" - выходной файл, в котором будет записан результат. Указывается имя файла с отчетом (имя, с которым в текущей папке будет сохранен файл с отчетом). Доступ к формату файла предоставляется по запросу. **Параметр задаётся "в кавычках"**.
3. -start_date "<start_date>" - дата начала периода персонализации - начальная дата периода, за который будет формироваться отчет. **Параметр задаётся "в кавычках"**.
4. -stop_date "<stop_date>" - дата окончания периода персонализации - конечная дата периода, за который будет формироваться отчет. **Параметр задаётся "в кавычках"**.
5. -part_num <part number> - part number (номер партии чипов), для которого нужно сформировать отчет.

6. -part_type <part type> - числовой идентификатор part type (типа партии чипов), для которого нужно сформировать отчет.

   ⚠ Один из параметров [-part_num <part number>] и [-part_type <part type>] является обязательным при запуске скрипта: либо должен быть указан <part number> (при этом <part type> не используется), либо - <part type> (при этом <part number> не используется).

7. -chip_start <chip start id> - начальный номер диапазона чипов - наименьший из диапазона ID чипов, данные о персонализации которого попадут в отчет.
8. -chip_stop <chip stop id> - конечный номер диапазона чипов - наибольший из диапазона ID чипов, данные о персонализации которого попадут в отчет.
9. -h <host> - имя хоста KTS DB. Значение по умолчанию (задается в файле скрипта) - "localhost".
10. -p <port> - номер порта KTS DB. Значение по умолчанию (задается в файле скрипта) - "5432".
11. -u <pg_user> - имя пользователя KTS DB. Значение по умолчанию (задается в файле скрипта) - "bbxadmin".
12. -P <pg_user_password> - пароль для доступа к KTS DB. Значение по умолчанию (задается в файле скрипта) - "bbxadmin".
13. -d <database> - имя базы KTS DB. Значение по умолчанию (задается в файле скрипта) - "bbx".

Формат команды запуска скрипта с параметрами:

```
bash blbx_make_report.sh [-help] -f "<output file>" [-start_date "<start_date>"] [-stop_date "<stop_date>"] [-part_num <part number>] [-part_type <part type>] [-chip_start <chip start id>] [-chip_stop <chip stop id>] [-hash tde hash file] [-h <host>] [-p <port>] [-u <pg_user>] [-P <pg_user_password>] [-d <database>]
```

Примеры:

- запуск скрипта с указанием part type:

  ```
  bash blbx_make_report.sh -part_type 131 -p 5433 -f "test_report"
  ```

- запуск скрипта с указанием part number:

  ```
  bash blbx_make_report.sh -part_num 1 -p 5433 -f "test_report"
  ```

Файл со сформированным отчетом зашифровывается PGP ключом администратора KTS и переносится на FTP-сервер KGS для последующего хранения.

### 5.1.4. Загрузка/удаление тестовых векторов на KTS сервере

⚠ Скрипт стоит запускать, только если созданы соответствующий part number (номер партии) и для этого part number OTP-ключи уже загружены в БД. Таким оборазом, процедуру по импортированию тестовых векторов следует выполнять только после загрузки соответствующих OTP-ключей на сервер KTS (см. выше).

Валидация ключей необходима для проверки 100% чипов, проходящих персонализацию на линии, сразу после записи ключей в OTP. Валидация направлена на проверку ключей из OTP. Как правило, само значение ключа из OTP недоступно для чтения в ПО, но может быть использовано чипом в криптоблоке или лестнице ключей, причем может использовать KDF-функцию. Поэтому валидация выполняется косвенным путем, через проверку на тестовых векторах. Тестовые вектора вычисляются в KGS и передаются в KTS вместе с самими OTP-ключами.

Скрипт import_OTP_validation_v5.py позволяет не только проверять и импортировать тестовые вектора в базу KTS, но удалять тестовые вектора для выбранного Part Type (Типа Партии) из БД, а также выполнять обе операции одновременно (сначала будет выполнено удаление, затем - импорт).

Необходимые условия:

- На сервере должно быть установлено следующее ПО:
  - python3 (третьей версии)
  - пакет psycopg2
- В системе KGS подготовлен файл с тестовыми векторами необходимого формата для партии чипов.

Установка необходимых пакетов:

1. С помощью pip (если установлен):

```
pip install psycopg2
```

2. С помощью apt-get:

```
apt install python3-psycopg2
```

Последовательность действий:

1. Файл с тестовыми векторами переносится с FTP-сервера KGS в рабочую папку БД KTS и расшифровывается PGP-ключом администратора.
2. Запускается скрипт *import_OTP_validation_v5.py* (входит в комплект поставки).
   В качестве аргументов скрипта указываются:
   a. -h DATABASE_HOST, --host DATABASE_HOST - имя хоста KTS DB. Значение по умолчанию (задается в файле скрипта) - 'localhost'.
   b. -p DATABASE_PORT, --port DATABASE_PORT - номер порта KTS DB. Значение по умолчанию (задается в файле скрипта) - '5432'.
   c. -u DATABASE_USER, --user DATABASE_USER - имя пользователя KTS DB. Значение по умолчанию (задается в файле скрипта) - 'postgres'.
   d. -P DATABASE_USER_PASSWORD, --pass DATABASE_USER_PASSWORD - пароль для доступа к KTS DB. Значение по умолчанию (задается в файле скрипта) - 'postgres'.
   e. -db DATABASE_NAME, --database DATABASE_NAME - имя базы KTS DB. Значение по умолчанию (задается в файле скрипта) - 'bbx'.
   f. -f FILENAME, --filename FILENAME - импорт файла с тестовыми векторами. Доступ к формату файла предоставляется по запросу.
   g. -pn PARTNUM, --partnum PARTNUM - Part Number name (наименование Номера Партии), для которого загружаются тестовые вектора (Part Number Name of validating devices).
   h. -v VERBOSE, --verbose VERBOSE - включение Verbose mode (debug вывод в консоль).

i. -d, --delete - удаление всех тестовых векторов из БД для указанного part type / part number (Типа Партии / Номера Партии).

3. Формат команды запуска скрипта с параметрами:

```
python3 import_OTP_validation_v5.py [-h DATABASE_HOST] [-p DATABASE_PORT] [-u DATABASE_USER] [-P
DATABASE_USER_PASSWORD] [-db DATABASE_NAME] [-f FILENAME] [-pn PARTNUM] [-v VERBOSE] [-d]
```

⚠ Обратите внимание:

- Поскольку скрипт может не только импортировать, но и удалять тестовые вектора, то параметр [-f FILENAME], как и остальные, является необязательным.
- Если задано -f, то происходит импорт, -d - удаление, оба - сначала удаление, потом импорт, -v - debug вывод в консоль.

4. Пример команды запуска скрипта с параметрами (импорт тестовых векторов):

```
python3 import_OTP_validation_v5.py -h 192.168.10.25 -p 5432 -u bbxadmin -P bbxadmin -db bbx -f "/home
/username/BBX/wizzards/24_21_30_personalization_data/otp_validation_pt_AML_processed.txt" -pn pn_AML -v
VERBOSE
```

5. Скрипт осуществляет загрузку тестовых векторов в целевые структуры БД.
6. При загрузке векторов в БД выполняется проверка ограничений (ограничения описаны в документе "Техническое описание" (доступ ограничен), в разделе "Скрипт загрузки тестовых векторов import_OTP_validation_v5.py"):
   a. Если какие-либо из условий не выполнены, скрипт прекращает работу и выводит в командную строку соответствующую ошибку.
   b. Если операция импорта была выполнена без ошибок, будет выведено много записей вида *INSERT INTO*.
7. (**Обязательно**) после импорта векторов в БД необходимо **перезагрузить** bbx_server_go.

### 5.1.5. Загрузка/удаление fusemap config file на KTS сервере

ⓘ FMC определяет полный список полей в OTP-памяти чипа с размерами и длинами, включая свойства OTP-ключей (без значений) и свойства и значения конфигурационных бит. Файл с FMC используется при персонализации чипа совместно с OTP-ключами.

Необходимые условия:

- Требуется получить (из KGS) следующие файлы:
  - файл с FMC (бинарный файл). При экспорте из KGS файл (fusemap config file) шифруется лестницей ключей требуемого KTS.
  - файл с хешом TDE KTS. Доступ к формату файла предоставляется по запросу.

Последовательность действий:

1. Необходимый(-ые) файл(-ы) переносится с FTP-сервера KGS в рабочую папку БД KTS и расшифровывается PGP-ключом администратора.
2. Запускается скрипт *blbx_load_fmc.sh*, входящий в комплект поставки.
   Формат команды запуска скрипта с параметрами:

```
bash blbx_load_fmc.sh [-help] -opt <option> -part_num <part number> [-fmc "<fmcd file>"] [-hash "<tde
hash file>"] [-h <host>] [-p <port>] [-u <pg_user>] [-P <pg_user_password>] [-d <database>]
```

Параметры запуска:

a. -help - вызов справки.
b. -opt option (a add/d delete) - выполняемая операция: *a* - добавление в базу; *d* - удаление. Обязательный параметр.
c. -part_num <part number name> - имя part number (Номера Партии), для которого загружается FMC. Обязательный параметр.
d. -fmc "<input fmc file>" - файл с fusemap config. Бинарный файл, зашифрованный по TDE (т.е. согласно лестнице ключей) для выбранного KTS. Доступ к формату файла предоставляется по запросу.
   **Параметр задаётся "в кавычках"** (т.е. если задан путь к файлу, то в кавычках указывается весь путь, если задано только имя файла (лежит в той же папке, что и скрипт), то в кавычках указывается имя файла).
e. -hash "<tde hash file>" - файл с хешом TDE KTS. Доступ к формату файла предоставляется по запросу.
   **Параметр задаётся "в кавычках"** (т.е. если задан путь к файлу, то в кавычках указывается весь путь, если задано только имя файла (лежит в той же папке, что и скрипт), то в кавычках указывается имя файла).
f. -h <host name> - имя хоста KTS DB. Значение по умолчанию (задается в файле скрипта) - "localhost".
g. -p <port> - номер порта KTS DB. Значение по умолчанию (задается в файле скрипта) - "5432".
h. -u <pg_user> - имя пользователя KTS DB. Значение по умолчанию (задается в файле скрипта) - "bbxadmin".
i. -P <pg_password> - пароль для доступа к KTS DB. Значение по умолчанию (задается в файле скрипта) - "bbxadmin".
j. -d <database> - имя базы KTS DB. Значение по умолчанию (задается в файле скрипта) - "bbx".

3. Скрипт осуществляет загрузку/удаление FMC в целевые структуры БД.
4. По окончании работы скрипта проверяется его лог файл (blbx_load_fmc.log) на отсутствие ошибок.

### 5.1.6. Кумулятивный импорт персонализационных данных на KTS сервере

Имеется возможность экспорта **единым (кумулятивным) архивом** из KGS данных, необходимых для персонализации чипов (OTP-ключи, конфигурация партии part type/part number, fusemap config, тестовые вектора для OTP-валидации).

С помощью скрипта cumulative_import.py осуществляется обработка, проверка и импорт этих данных в базу KTS (иными словами, "кумулятивный импорт"либо "импорт кумулятивного архива").

Скрипт запускается под любым пользователем.

Поскольку cumulative_import.py при своей работе дублирует либо вызывает другие скрипты (blbx_load_config.sh, blbx_load_keys.sh, import_OTP_validation_v5.py, blbx_load_fmc.sh), то можно сказать, что на cumulative_import.py суммарно налагаются все их требования и ограничения (кроме входных файлов).

Необходимые условия:

- Требуется получить (из KGS) и расшифровать (PGP) кумулятивный архив. Имя общего архива: *<PartTypeID>_<Start Device Number>_<End Device Number>_personalization_data.zip*
- На сервере должно быть установлено следующее ПО:
  - python3 (третьей версии)

    > python3 входит в состав эталонного образа ОС ("устанавливается из коробки").

  - пакет python3-pip (для установки simple_file_checksum)
  - пакет simple_file_checksum
  - пакет psycopg2

Установка необходимых пакетов:

1. Установить pip (если не установлен):

```
sudo apt-get install python3-pip
```

2. Установить simple_file_checksum (до использования cumulative_import.py):

```
pip install simple_file_checksum
```

3. Установить psycopg2:
   a. С помощью pip:

```
pip install psycopg2
```

   b. С помощью apt-get:

```
sudo apt-get install python3-psycopg2
```

Последовательность действий:

1. Необходимый архив переносится с FTP-сервера KGS в рабочую папку БД KTS и расшифровывается PGP-ключом администратора.
2. Запускается (под любым пользователем) скрипт *cumulative_import.py* (входит в комплект поставки). Формат команды запуска скрипта с параметрами:

```
python3 cumulative_import.py -a "CUMULATIVE_ARCHIVE" [-h DATABASE_HOST] [-p DATABASE_PORT] [-u
DATABASE_USER] [-P DATABASE_USER_PASSWORD] [-d DATABASE_NAME] [-s DATABASE_SCHEMA]
```

Параметры запуска:

a. -a "CUMULATIVE_ARCHIVE", --archive "CUMULATIVE_ARCHIVE" - кумулятивный архив. Обязательный параметр.
**Параметр задаётся "в кавычках"** (т.е. если задан путь к файлу, то в кавычках указывается весь путь, если задано только имя файла (лежит в той же папке, что и скрипт), то в кавычках указывается имя файла).
b. -h DATABASE_HOST, --host DATABASE_HOST - имя хоста KTS DB. Значение по умолчанию (задается в файле скрипта) - 'localhost'.
c. -p DATABASE_PORT, --port DATABASE_PORT - номер порта KTS DB. Значение по умолчанию (задается в файле скрипта) - '5432'.
d. -u DATABASE_USER, --user DATABASE_USER - имя пользователя KTS DB. Значение по умолчанию (задается в файле скрипта) - 'postgres'.
e. -P DATABASE_USER_PASSWORD, --pass DATABASE_USER_PASSWORD - пароль для доступа к KTS DB. Значение по умолчанию (задается в файле скрипта) - 'postgres'.
f. -d DATABASE_NAME, --database DATABASE_NAME - имя базы KTS DB. Значение по умолчанию (задается в файле скрипта) - 'bbx'.
g. -s DATABASE_SCHEMA, --schema DATABASE_SCHEMA - схема базы KTS DB.

3. Пример команды запуска:

```
python3 cumulative_import.py -a "/home/user/kgs/wizard/19_113_113_personalization_data.zip" -h
192.168.10.25 -p 5432 -u bbxadmin -P bbxadmin -d bbx -s bbx
```

4. Скрипт осуществляет проверку и загрузку персонализационных данных в целевые структуры БД.
5. По окончании работы скрипта проверяются лог файлы скриптов, вызываемых из cumulative_import.py, на отсутствие ошибок.
6. (**Обязательно**) после импорта векторов в БД необходимо **перезагрузить** BBX_CHIP_SERVER.

# 6. Ведение Логов

## 6.1. Режимы Ведения Логов

Компонент ведет логи, информация из которых может быть использована для решения возникающих проблем. Логи могут вестись с разной степенью подробности.

Доступны следующие режимы ведения логов:

- 0 - trace: подробная информация по любым действиям;
- 1 - debug: конфигурационные данные (при запуске системы), другая информация, необходимая для отладки, + сообщения уровня Info;
- 2 - info (значение по умолчанию): базовая информация (сообщения о запуске, работе, выключении системы) + сообщения уровня Warning;
- 3 - warning: системные предупреждения + сообщения уровня Error;
- 4 - error: все ошибки, возникающие в процессе работы, в том числе ошибки уровня Fatal;
- 5 - fatal: критические ошибки, приводящие к сбоям системы.

## 6.2. Форматы логов

Все логи представлены в формате JSON. Каждое сообщение лога представлено в формате отдельной JSON-структуры с исчерпывающим набором полей.

В зависимости от уровня логирования изменяется набор данных внутри JSON либо добавляются новые элементы (записи лога).

Логи для режима **debug** имеют структуру JSON и по формату делятся на следующие виды:

1. Лог для event: "request";
2. Лог для event: "response";

Логи для event: "request" и event: "response" содержат информацию по запросу к Системе и ответу на запрос соответственно.

Описание логов выполнено следующим образом:

- Пример лога указанного типа.
- Таблица с описанием параметров, используемых только в логе данного типа. Указываются только те параметры, которые являются более-менее частными (уникальными) для данного лога (остальные параметры приведены в разделе "Общие параметры"). Если у формата лога нет уникальных параметров, то таблица отсутствует.

### 6.2.1. Общие параметры

Общие параметры, которые встречаются во всех типах логов, приведены в таблице ниже.

| Параметр | Описание |
|---|---|
| app | Внутреннее наименование приложения (микросервиса KTS) |

| | |
|---|---|
| app_ver | Версия приложения (микросервиса KTS) |
| cid | Correlation id, используемый для трассировки запросов |
| "details": | Дополнительные передаваемые параметры (состав различается от одного формата лога к другому) |
| event | Тип логируемого события. От типа логируемого события зависит формат лога |
| level | Уровень логирования (см. Режимы Ведения Логов)<br><br>**Примечание**. Структура лога регламентирована только для "level":"debug" |
| msg | Комментарий к логу |
| time | Дата и время (вплоть до миллисекунд) формирования записи (фиксации лога) |

### 6.2.2. event: "request"

**Пример лога для event: "request"**

```
{
        "app":"bbx_server",
        "app_ver":"1.0.0-dev.24",
        "cid":"443c0a8d-3adf-4d5b-aba5-735740e6a0a7",
        "details":{
                "body":"",
                "headers":{
                        "Accept":["*/*"],
                        "Content-Type":["application/json"],
                        "Grpc-Metadata-Cid":["443c0a8d-3adf-4d5b-aba5-735740e6a0a7"],
                        "X-Correlation-Id":["443c0a8d-3adf-4d5b-aba5-735740e6a0a7"],
                        "X-Forwarded-For":["test"]
                },
                "method":"GET",
                "path":"/api/v1/part_numbers/pn1_test/devices/keys?amount=1",
                "protocol":"HTTP/1.1",
                "request_length":0
        },
        "event":"request",
        "level":"debug",
        "msg":"Request accepted",
        "time":"2023-01-31T19:24:56.747+03:00"
}
```

| Параметр | Описание |
|---|---|
| details.body | Тело запроса<br><br>Опционально может иметь null, "", {} |
| details.headers | BCE headers (заголовки) в запросе (т.е. дополнительно может содержать cookies, X-correlation-id и т.п.) |
| details.method | Метод (тип) http-запроса |
| details.path | URL запроса |

| | |
|---|---|
| details.protocol | Протокол взаимодействия<br><br>Значение по умолчанию: "HTTP/1.1" |
| details.<br>request_length | Длина запроса (в Байтах) |

### 6.2.3. event: "response"

**Пример лога для event: "response"**

```
{
        "app":"bbx_server",
        "app_ver":"1.0.0-dev.24",
        "cid":"443c0a8d-3adf-4d5b-aba5-735740e6a0a7",
        "details":{
                "body":{
                        "keys":[
                                {
                                        "device_id":108,
                                        "key_data":"",
                                        "part_type_id":5
                                }
                        ]
                },
                "headers":{
                        "Content-Type":["application/json"],
                        "X-Correlation-Id":["443c0a8d-3adf-4d5b-aba5-735740e6a0a7"]
                },
                "status":200
        },
        "event":"response",
        "level":"debug",
        "msg":"Response was sent",
        "time":"2023-01-31T19:24:56.752+03:00"
}
```

| Параметр | Описание |
|---|---|
| details.body | Тело запроса<br><br>Опционально может иметь null, "", {} |
| details.body.<br>errors | Описание ошибки (код, заголовок, текст ошибки)<br><br>**Примечание**. Передается, только если запрос завершился ошибкой |
| details.body.<br>headers | ВСЕ headers (заголовки) в ответе (т.е. дополнительно может содержать cookies, X-correlation-id и т.п.) |
| details.status | Код ответа от сервера |

## 6.3. Логи BBX_CHIP_CLIENT

Клиентская библиотека пишет логи ошибок (с пояснениями, если из кода ошибки причина не понятна) в bbx_chip_client_errors.log