GS LABS
GS GROUP

Публичный - L0

Программный комплекс «Система условного доступа DREGUARD»

Руководство по установке

Индекс	CAS-DREGUARD-IG
Конфиденциальность	Публичный - L0
Ревизия	1.1
Статус	Согласован

Содержание

1. Аннотация

Данный документ содержит руководство по установке и первоначальной настройке Программного комплекса "Система условного доступа DREGUARD" (далее - CAS DREGUARD или Система), а также описание системных требований для компонентов.

Документ предназначен для сотрудников отдела мониторинга и инсталляции, а также для других технических специалистов, в обязанности которых входит установка и первоначальная настройка CAS DREGUARD.

Данное описание является публичным документом.

2. Термины и сокращения

Термин	Сокращение	Определение
Абонент	-	Физическое или юридическое лицо, с которым оператор ТВ заключает договор на оказание услуг.
Авторизация	-	Процесс предоставления абоненту прав на использование услуг телевизионного оператора (просмотр телевизионных каналов и услуги интерактивных телевизионных сервисов).
Антишаринг	-	Подход, используемый в разработке технологий по противодействию нелегальному доступу к каналам, закрытым CAS DREGUARD, при котором распространяются ключи (CW), полученные от одной авторизованной смарт-карты.
Биллинговая система (Business Support System)	BSS	Сторонний по отношению к CAS DREGUARD компонент, отвечающий за сбор информации об использовании услуг, выставление счетов абонентам, обработку платежей. На основании этой информации биллинговая система выдает CAS DREGUARD команды на добавление, удаление или изменение подписок, сообщений Инфокас, или иных данных, хранящихся в CAS DREGUARD.
Головное оборудование	-	Оборудование головной станции оператора. В данном документе под головным оборудованием подразумевается та его часть, которая непосредственно взаимодействует с Системой Условного Доступа согласно стандарту DVB-SimulCrypt.
Класс	-	Единица доступа в CAS DREGUARD. С точки зрения пользователя CAS DREGUARD (оператора ТВ), класс определяет пакет телевизионных каналов, предоставляемых абоненту в качестве единой услуги. На базе управления правами на класс реализовано управление доступом абонента к пакету каналов. При оформлении подписки на класс, абоненту предоставляется доступ ко всем телевизионным каналам, входящим в его состав.
Контрольное слово (Control word)	CW	Ключи, используемые для скремблирования/дескремблирования транспортного потока алгоритмом CSA.
ПАК "Криптокипер" (Криптокипер, Cryptokeeper)	CK	Специализированное шифрующее устройство.
Криптопериод		Период времени, в течение которого скремблером используется один и тот же ключ скремблирования (CW).

Мастер-ключ	MK	Ключ, необходимый для декодирования операционных ключей, получаемых из EMM сообщений. Мастер-ключ относится к самому верхнему уровню иерархии ключей и хранится в самой защищённой области энергонезависимой памяти смарт-карты. Этот ключ получается легальным пользователем вместе со смарт-картой и никогда не меняется.
Оператор ТВ (TV Provider)	-	Организация, предоставляющая услуги просмотра цифрового телевидения и использования дополнительных сервисов.
Операционный ключ	OPKEY	Ключ, используемый для шифрования и расшифровывания управляющих слов (CW). Операционные ключи передаются в зашифрованном виде в составе специальных команд в EMM сообщениях.
Пакет услуг	-	Данный термин используется в рамках функционала WHN. Совокупность прав на использование услуг оператора ТВ, предоставляемая абоненту при заключении договора. Подписка на пакет услуг определяет доступные для абонента типы услуг (Streaming, PVR и прочее), классы подключаемых устройств (например, планшеты GS), максимальное количество одновременно подключаемых устройств. Пакет услуг может иметь привязку к одному или нескольким каналам.
Подписка	-	Информация о правах доступа абонента к классам и услугам оператора ТВ (идентификатор класса, идентификатор пакета услуг и период, на который они предоставлены).
Приёмник (Set Top Box)	STB	Устройство абонента, принимающее и обрабатывающее сигнал цифрового телевидения и передающее его далее для воспроизведения (например, на телевизоре или планшете).
Оператор ТВ	-	Организация, предоставляющая услуги просмотра цифрового телевидения и использования дополнительных сервисов.
Система управления подписками (Subscriber Management System)	SMS	Система, принимающая, обрабатывающая и хранящая информацию о подписках абонентов.
Скремблер	SCR	Устройство шифрования транспортного потока, входящее в состав головного оборудования. В терминологии стандарта DVB-Simulcrypt обозначает функциональный логический блок, ответственный за шифрование MPEG2 транспортного потока. Для выполнения данной функции должен обеспечивать прием CW от компонента SCS.
Смарт-карта	-	Версия внешней смарт-карты, реализованная на аппаратной платформе. Используется в CAS DREGUARD для идентификации пользователей, безопасного хранения приватных данных и защиты криптографических операций, необходимых для управления доступом.

Стриминг (Streaming)	-	Функция приёмника, позволяющая передавать контент с приёмника-сервера на устройство-клиент (приёмник, планшет). Различают следующие виды Streaming: • Mirror Streaming: на мобильное устройство транслируется контент, воспроизводимый на сервере; • Independent Streaming: на мобильное устройство транслируется контент по запросу клиента, независимо от контента, воспроизводимого на сервере.
Услуга	-	Дополнительная платная функциональность приёмного оборудования, доступ к которой ограничивается оператором цифрового телевидения в соответствии с разработанными им правилами использования данной функциональности (например, использование Streaming и т.д.).
Access criteria (Критерий доступа)	AC	Данные системы условного доступа, необходимые ECMG для формирования ECM сообщений. Состав и структура этих данных определяется разработчиком CAS DREGUARD и является закрытой информацией
Conditional access system (Система условного доступа)	CAS	Система условного доступа обеспечивает защиту контента, передаваемого по каналам вещания и распределения, от коммерческого пиратства.
CAS DREGUARD Library (далее по тексту - Library)	-	Система, представляющая собой промежуточное ПО, которое обеспечивает взаимодействие ПО приемника и эмулятора смарт-карты. Library управляет выделением служебных данных (например, ECM и EMM) из принятого транспортного потока, получением из них зашифрованных ключей и их передачу эмулятору смарт-карты для расшифровки.
DRE Account Manager (далее по тексту - Account Manager)	ACM	Сервис авторизации и распределения прав. CAS DREGUARD может обратиться к Account Manager (через свой фронтенд или API) для авторизации, передачи и проверки прав пользователя. Затем, при помощи WEB UI, пользователь CAS DREGUARD может создавать дополнительные учетные записи, назначать роли и создавать группы прав.
DRE Advanced Encryption Service	-	Программа, обеспечивает возможность дополнительной защиты контента при вещании в спутниковой и IP сетях.
DRE Config Manager (далее по тексту - ConfigManager)	CM	Сервис для хранения списка операторов и их конфигурационных данных. Сервис получает, агрегирует и выдает перечень операторов, которые используют DREAMPlatform, а также набор конфигурационных данных для каждого оператора. Таким образом, абонент, пользуясь одним клиентским приложением, может получать доступ к нескольким операторам.

DRE Messaging Service (далее по тексту - Hermes)	-	Система, реализующая функционал отправки уведомлений на оборудование абонентов.
DVB-Simulcrypt	-	DVB-стандарт архитектуры, позволяющей функционировать множеству Систем Условного Доступа в рамках единой головной станции. Этот стандарт определяет архитектуру головного оборудования и СУД, временные параметры взаимодействия компонентов, их интерфейсы и формат сообщений.
Electronic Program Guide, (электронный программный гид)	EPG	Электронный телегид. Сервер генерирует поток данных с файлами расписания телепередач, расписания показа фильмов и файлами обновления ПО, встроенного в приёмники.
Encoded Channel DRE	ECD	Дополнительная функция CAS DREGUARD, предназначенная для информирования абонента о причине ограничения доступа к телепередаче. Информирование производится через сообщения "Кодированный канал", которые содержат код статуса дескремблирования и краткую инструкцию по устранению проблемы.
Enhanced Common Scrambling Algorithm	ECSA	Технология аппаратного антишаринга, построенная на дополнительном шифровании CW до его шифрования алгоритмом смарт-карты. Требует аппаратной поддержки в процессоре STB, в который встраивается библиотека (CAS DREGUARD Library).
Entitlement Control Message	ECM	Сообщение CAS DREGUARD, содержащее в зашифрованном виде CW, дескремблирующие транслируемый поток.
Entitlement Management Message	EMM	Сообщение CAS DREGUARD, содержащее служебные данные, информацию о правах доступа и специальные команды (изменение подписки, обновление операционного ключа и другие).
InfoCAS	-	Дополнительная функция CAS DREGUARD, позволяющая оператору ТВ рассылать текстовые сообщения абонентам. Функция реализуется системой CAS DREGUARD или системой Сервисов. Сообщения принудительно отображаются на экране ТВ поверх основного изображения. Примеры использования сообщений - предупреждения об обновлении ПО, окончании подписки, оповещения населения и другие.
Lite mode	-	Режим работы ECMG. В данном режиме CW передаются в ECM в незашифрованном виде.

L-режим	-	Режим ограниченного просмотра, предназначенный для мотивации абонента продлить подписку. В данном режиме картинка телеканала пропорционально уменьшается, а в остальной области экрана выводится реклама.
Pairing	-	Режим CAS DREGUARD, обеспечивающий работу смарт-карты только с одним конкретным приёмником.
Scrambling Control Group	SCG	Логическое объединение каналов, скремблируемых едиными CW, создаваемое на головном оборудовании Оператора ТВ.
Simulcrypt синхронизатор	SCS	Компонент головного оборудования, предназначенный для установления и поддержания соединения с ECMG, передачи ему CW и AC, получения сгенерированных ECM сообщений и перенаправление их в MUX.
Мультиплексор (Multiplexer)	MUX	Компонент головного оборудования, предназначенный для преобразования получаемой информации в TS с последующей передачей на спутник.
Common Scrambling Algorithm	CSA	Общий алгоритм скремблирования, используемый для защиты цифрового телевизионного потока от несанкционированного доступа
Transport Stream	TS	Формат медиаконтейнера, который инкапсулирует пакеты элементарных потоков и других данных.
TS Monitor	TSM	Набор программных средств, предназначенных для мониторинга транспортного потока.
Subscriber Management System	SMS	Компонент который позволяет управлять подписками/услугами /устройствами/каналами и т.д. Обеспечивает интеграцию с биллинг системами операторов.

3. Введение

3.1. Требования к квалификации установщика

Для установки системы сотрудник обязан:

- иметь базовые представления и практические навыки работы с системой оркестрации Kubernetes (https://kubernetes.io/docs/tutorials/kubernetes-basics/) и пакетным менеджером Helm.
- иметь навыки работы с ОС семейства Linux, а именно:
    - установка пакетов;
    - создание и настройка сетевых подключений;
    - запуск служб, настройка автозапуска служб;
    - установка и настройка PostgreSQL;
    - создание и работа с БД под управлением PostgreSQL.
- иметь знания о DNS.
- иметь базовые представления и практические навыки работы с Git.

3.2. Системные Требования

Для установки CAS DREGUARD желательно выделить отдельный сервер. Рекомендуется устанавливать сервер в локальной сети, защищенной от доступа извне.

3.2.1. Аппаратное Обеспечение

- Процессор — 4 ядра;
- Оперативная память — зависит от размера базы абонентов, с которой будет работать CAS DREGUARD. Минимум 8 GB;
- Жесткий диск — 2 × 150 GB (зависит от объема БД);
- Головное оборудование, соответствующее стандарту DVB-Simulcrypt ver. 2.

3.2.2. Программное Обеспечение

- Компоненты CAS DREGUARD поставляются в контейнерах, поэтому основное требование к ОС сервера - возможность установить Docker.
- Предполагается, что базы данных (OPKEY DB, CAS DB, Carousel DB и т.д.) будут развернуты на *nix системе.

3.2.3. Системные требования для развертывания компонентов Системы

Компоненты Системы разворачиваются в кластере Kubernetes. Для данных компонентов должна быть развернута одна нода кластера.

Для установки необходимо предварительно выполнить следующие требования:

- На отдельном сервере подготовлена Ansible node с поддержкой CI/CD. За информацией обращаться к разработчику платформы автоматизации CI/CD ООО "Цифра".
- Установлен и настроен кластер Kubernetes через K3s.
    - Так как развертывание производится в кластере k8s, то необходим config file для доступа к кластеру.

1. Если пользователь выполнял развертывание Kubernetes самостоятельно, то он сам должен создать config file (см. документацию Kubernetes).
2. Если Kubernetes был развернут сторонними людьми, то необходимо получить config file у администратора кластера.

- На машине администратора установлен kubectl (https://kubernetes.io/docs/tasks/tools/install-kubectl/).
- На машине администратора установлен helm.
- Развернут DNS-сервер, преобразование имен dns зоны настроено на мастера k8s (созданы A записи на зону dns). DNS устанавливается в сетевое окружение DMZ зоны, где будет развернут CAS DREGUARD.
- Для корректной работы системы CAS DREGUARD требуется развернуть кластер БД (ссылка и права доступа к инструкции по развертыванию кластера БД предоставляются по запросу заказчика)
- Для корректной работы системы CAS DREGUARD необходим доступ к следующим ресурсам:
    - chartmuseum (ссылка и права доступа предоставляются по запросу заказчика)
- Необходим доступ к репозиторию, содержащему helmfile для развертывания CAS. Helm файл содержит инструкции, с помощью которых осуществляются настройки устанавливаемых компонентов. Сами компоненты поставляются в виде образов (images), из которых разворачиваются Docker-контейнеры.

4. Состав компонентов для установки системы

4.1. Компоненты CAS DREGUARD

Развертывание компонентов CAS DREGUARD осуществляется из репозитория gitlab (с помощью CI/CD). Необходимые сборки, если не указано иное, лежат в gitlab.

Репозиторий, содержащий helmfile для развертывания CAS DREGUARD.

4.2. RCAS scripts

Скрипты по работе с базами данных.

Поставляются в виде архива ***rcas_scripts_X.X.X.zip***

5. Установка компонентов CAS DREGUARD

5.1. Процедура установки

Необходимо выполнить установку системы в соответствии с документом "Описание схемы CD", тэг 4.0 (ссылка и права доступа предоставляются по запросу заказчика).

5.2. Как создать новую среду

1. Создать отдельный проект в Gitlab
2. Настроить проект certification/rcas как подмодуль на основе инструкции (ссылка и права доступа предоставляются по запросу заказчика)
3. В проекте среды создать helmfile.yaml с содержимым:

```
---
helmfiles:
  - path: <путь до подмодуля>/helmfile.yaml
    values:
      - <путь до подмодуля>/default.yaml # Загружаем зачения по-умолчанию
      - production.yaml                  # Применяем собственную конфигурацию
      - versions.yaml                    # (опционально) Переопределяем версии некоторых компонентов
```

5.3. Пример .gitlab-ci.yml

```
# здесь перечисляются необходимые шаги(stage) пайплайна
# в случае, если часть вышеописанного функционала
# не требуется, ненужные шаги можно удалить
# (например, оставить только init)
stages:
  - init
  - compose
  - grade

variables:
  # GIT_* переменные необходимы для правильной работы
  # репозитория с сабмодулем
  GIT_SUBMODULE_STRATEGY: recursive
  GIT_STRATEGY: clone
  # если namespace релиза не задаётся через values/шаблоны/helmfile,
  # то его можно задать через переменную NAMESPACE
  NAMESPACE: rcas-stand
  STAGED_PIPELINE: "true"

include:
  - project: 'automation/cd-templates'
    ref: "4.0"
    file: pipeline.yml
```

5.4. Двухступенчатый деплой

Для выполнения двухступенчатого деплоя, в случае если часть релизов, описанных в helmfile, следует установить прежде остальных, следует выполнить три условия:

- задать в файле .gitlab-ci.yml переменную *STAGED_PIPELINE* в значение *true*;
- в helmfile.yaml задать переменные *wait* и *waitForJobs*;
- указать для каждого релиза этап его установки посредством меток *stage: first* или *stage: second*.

При этом возможно так же установить допустимый период ожидания выполнения установки релизов/джобов посредством переменной timeout (по умолчанию - 300).

Версия шаблонов CI должна быть не менее 4.0.

5.5. Настройка и развертывание CAS DREGUARD

5.5.1. CD для артефактов БД

При развертывании CAS DREGUARD происходит установка SCH и API для БД через механизм Kubernetes Jobs. В процессе установки сохраняется лог в контейнере.

```
sms_db_sch:
  enabled: true
  # You can optionally override database address and port here:
  #db:
  #  address: 127.0.0.1
  #  port: 5432

sms_db_api:
  enabled: true
```

Этот режим поддерживают все базы данных системы CAS DREGUARD.

5.5.2. Настройка переменных окружения

В системе развертывания CAS DREGUARD требуется указывать переменные окружения, которые используются непосредственно в самом процессе деплоя CAS DREGUARD в кластер.

Настройка переменных осуществляется в gitlab.

В боковом меню выбрать **Settings** (на панели слева) -> **CI/CD** -> **Environment variables**. Отредактировать переменные.

Таблица с описанием используемых переменных Gitlab

Переменная	Описание
ERRMAPDB_LOGIN	Имя пользователя для Errmap БД
ERRMAPDB_PASSWORD	Пароль пользователя для Errmap БД
POSTGRES_LOGIN	Имя администратора PostgreSQL БД
POSTGRES_PASSWORD	Пароль администратора PostgreSQL БД
SMSDB_LOGIN	Имя пользователя для SMS БД
SMSDB_PASSWORD	Пароль пользователя для SMS БД

CAROUSELDB_LOGIN	Имя пользователя для CAROUSEL БД
CAROUSELDB_PASSWORD	Пароль пользователя для CAROUSEL БД
CASDB_LOGIN	Имя пользователя для CAS БД
CASDB_PASSWORD	Пароль пользователя для CAS БД
OPKEYDB_LOGIN	Имя пользователя для OPKEY БД
OPKEYDB_PASSWORD	Пароль пользователя для OPKEY БД

ВАЖНО! Environment variables имеют более высокий приоритет, чем переменные, заданные в файлах.

Параметры _LOGIN и _PASSWORD задаются пользователем и используются при подключении к соответствующим базам данных.

Обратите внимание! Те же значения должны быть указаны при настройке PostgreSQL (см. раздел Настройка PostgreSQL). Значения по умолчанию (имена пользователей) для баз данных приведены в разделе Настройка PostgreSQL.

5.5.3. Настройка additional

Папка ***additional*** содержит файлы, с помощью которых настраиваются dns, ingress, probes, statsd. Указанные параметры применяются ко всем сервисам и службам в данном репозитории. **Рекомендуется не менять эти настройки**.

5.5.4. Состав репозитория

Репозиторий имеет следующий состав:

- helmfile.yaml - главный конфигурационный файл утилиты helmfile.
- default.yaml - файл с values окружения утилиты helmfile.
- values - папка с values для каждого чарта; они являются шаблонными и забирают значения из values окружения (файла default.yaml).
- versions.yaml - файл с версиями компонентов; если в версии установлена пустая строка, то берется последняя версия (в соответствии с semver2).
- limitation - папка с values ресурсов подов. С помощью этих файлов настраиваются компоненты CAS DREGUARD, в том числе многочисленные базы данных.

5.5.5. Выбор компонентов CAS DREGUARD для установки

По умолчанию разворачиваются все компоненты CAS DREGUARD, однако при необходимости можно отключать ненужные: для этого в production.yaml, в корне секции соответствующего компонента нужно выставить *enabled: false*.

5.5.6. Динамические параметры в конфигурационных файлах

В конфигурационных файлах *_server.cfg параметры разделены на две группы:

1. Все параметры, лежащие вне секции "system". Эти параметры можно менять динамически, т.е. без перезапуска соответствующей службы. При изменении значений этих параметров в конфигурационном файле, по прошествии некоторого времени, новые значения будут автоматически применены к службе.

   ⚠ **Обратите внимание!** Параметры, изменяемые динамически, нельзя задать через переменные окружения (см. выше), они меняются только в конфигурационном файле.

2. Параметры в секции "system". Эти параметры нельзя изменить динамически: чтобы изменения этих параметров вступили в силу, соответствующая служба должна быть перезапущена.

Некоторые из динамически изменяемых параметров (например, xxx.host в *.cfg) нельзя применять со значениями "по умолчанию", они должны быть настроены на production.

5.6. Развертывание системы (основные этапы)

Этапы развёртывания:

1. Подготовка данных в git (см. гл. "Установка компонентов CAS DREGUARD"):
   a. Изучить документ "Описание схемы CD" (ссылка и права доступа предоставляются по запросу заказчика). Выполнить описанные процедуры.
   b. Настроить двухступенчатый деплой.
   c. Настроить environment variables (см. "Настройка переменных окружения").
   d. Настроить yml-файлы, которые определяют состав и настройки разворачиваемых сервисов и баз данных, см. "Настройка additional", "Состав репозитория".
   e. В конфигурационных файлах настроить параметры, которые нельзя оставлять "по умолчанию" и /или нельзя изменить динамически, см. "Динамические параметры в конфигурационных файлах ".
2. Установка баз данных, входящих в состав Системы, в git (с помощью CI/CD). См. "Установка Баз Данных ".

   ⚠ Развертывание CAS DREGUARD происходит в два этапа: сначала производится установка баз данных, затем (после их наполнения) - сервисов CAS DREGUARD.

   Этапы (stages) настраиваются в gitlab-ci.yaml.

   a. Перед установкой требуется создать и соответствующим образом настроить production.yaml (см. CD для артефактов БД).
3. Наполнение баз данных (с помощью скриптов). См. "Установка Баз Данных".
   a. **Обязательно** выполнить скрипт individual_keys_get.sh (см. "Настройка кэширования индивидуальных ключей").
4. Установка служб/сервисов, входящих в состав Системы, в git (с помощью CI/CD).
   a. Перед установкой требуется создать и соответствующим образом настроить production.yaml.
   b. Делается несколько инстансов CAS DREGUARD:
      i. один - для sms_postgres

ii. по одному инстансу CAS DREGUARD - для каждого провайдера (provider_id)

5. **(Обязательно) удалить jobs**, созданные при развертывании баз данных, иначе в дальнейшем нельзя будет накатить новые DB_API и DB_SCH.

> ⚠ **ВНИМАНИЕ!** При установке в production базы (XXX DB) её старые схемы XXX_DB_API, соответствующие более ранним релизам, автоматически не удаляются. Т.е. старые схемы XXX_DB_API нужно удалять вручную.

5.7. Редактирование production.yaml (для сервисов CAS DREGUARD)

Файл production.yaml создается на основе default.yaml (ссылка предоставляется по запросу заказчика), содержащего основные настройки Системы. Настройки, заданные в default.yaml, кроме (опционально) параметров подключения, являются достаточными для эксплуатации Системы.

Особенности:

- Значения параметров, заданные в production.yaml, имеют более высокий приоритет, чем значения, заданные в default.yaml.
- Если параметр не задан в production.yaml, то будет использовано значение, заданное в default.yaml.
- Если параметр не задан ни в production.yaml, ни в default.yaml, то будет использовано значение, заданное в конфигурационном файле данного компонента.

6. Предварительные действия

6.1. Установка ОС

1. Установите на сервере желаемую ОС. Следуйте стандартный инструкциям по настройке данной ОС. Предполагается, что сервер будет функционировать под управлением ОС *nix.
2. Установите Docker на данной ОС.

6.2. Настройка локализации

Проверьте, что у вас активна локаль **ru_RU.UTF-8**. Например, в Debian это можно сделать так:

1. Выполните команду:

```
sudo dpkg-reconfigure -plow locales
```

2. Убедитесь, что в списке локализаций отмечена **ru_RU.UTF-8**. Если это не так, выберите её в добавок к уже имеющимся и нажмите *Ok*.
3. Проверьте, что вывод имеет вид:

```
Generating locales (this might take a while)...
   en_US.UTF-8... done
   ru_RU.UTF-8... done
Generation complete.
```

6.3. Установка PostgreSQL

По умолчанию требуется развернуть кластер БД (ссылка и права доступа к инструкции по развертыванию кластера БД предоставляются по запросу заказчика)

Данный раздел следует использовать только в случае установки БД в режиме Standalone.

Для работы системы CAS DREGUARD требуется PostgreSQL версии 14 или выше.

Ниже приведен пример установки PostgreSQL на сервер без развертывания и настройки кластера БД.

1. (Рекомендуется) обновить текущие системные пакеты, если это новый экземпляр сервера:

```
sudo apt update
sudo apt -y install vim bash-completion wget
sudo apt -y upgrade
```

   Установите дополнительные пакеты (локаль):

```
locale -a
sudo locale-gen ru_RU.UTF-8
sudo dpkg-reconfigure locales
```

Выполните перезагрузку:

```
sudo reboot
```

2. Добавьте репозиторий PostgreSQL 14:
    a. Перед настройкой репозитория APT импортируйте ключ GPG, используемый для подписи пакетов:

    ```
    sudo apt update
    sudo apt -y install gnupg2
    wget --quiet -O - https://www.postgresql.org/media/keys/ACCC4CF8.asc | sudo apt-key add -
    ```

    b. После импорта ключа GPG добавьте содержимое репозитория в ОС:

    ```
    echo "deb http://apt.postgresql.org/pub/repos/apt/ `lsb_release -cs`-pgdg main" |sudo tee  /etc
    /apt/sources.list.d/pgdg.list
    ```

    c. Добавленный репозиторий содержит много различных пакетов, включая сторонние дополнения. Они включают:

        i. PostgreSQL-клиент
        ii. PostgreSQL
        iii. libpq-DEV
        iv. PostgreSQL-сервер-DEV
        v. пакеты pgadmin

    d. Cat файл, созданный для проверки его содержимого:

    ```
    $ cat /etc/apt/sources.list.d/pgdg.list
    deb http://apt.postgresql.org/pub/repos/apt/ buster-pgdg main
    ```

3. Установка пакетов PostgreSQL 14:
    a. Обновите список пакетов и установите серверные и клиентские пакеты PostgreSQL 14:

    ```
    sudo apt update
    sudo apt -y install postgresql-14 postgresql-client-14 postgresql-14-cron
    ```

    b. Запустите сервер базы данных, используя следующую команду:

    ```
    sudo pg_ctlcluster 14 main start
    ```

    c. Подтвердите статус службы и используемый файл конфигурации:

```
$ sudo pg_ctlcluster 14 main status
pg_ctl: server is running (PID: 4209)
/usr/lib/postgresql/14/bin/postgres "-D" "/var/lib/postgresql/14/main" "-c" "config_file=/etc
/postgresql/14/main/postgresql.conf"
```

d. Можно использовать команду *systemctl* для проверки статуса службы. В случае успешной установки выводится сообщение примерно следующего вида:

```
$ systemctl status postgresql.service
  postgresql.service - PostgreSQL RDBMS
    Loaded: loaded (/lib/systemd/system/postgresql.service; enabled; vendor preset: enabled)
    Active: active (exited) since Sun 2019-10-06 10:23:46 UTC; 6min ago
  Main PID: 8159 (code=exited, status=0/SUCCESS)
     Tasks: 0 (limit: 2362)
    CGroup: /system.slice/postgresql.service
 Oct 06 10:23:46 debian systemd[1]: Starting PostgreSQL RDBMS…
 Oct 06 10:23:46 debian systemd[1]: Started PostgreSQL RDBMS.

$ systemctl status [email protected]
  [email protected] - PostgreSQL Cluster 14-main
    Loaded: loaded (/lib/systemd/system/[email protected]; indirect; vendor preset: enabled)
    Active: active (running) since Sun 2019-10-06 10:23:49 UTC; 5min ago
  Main PID: 9242 (postgres)
     Tasks: 7 (limit: 2362)
    CGroup: /system.slice/system-postgresql.slice/[email protected]
            9242 /usr/lib/postgresql/14/bin/postgres -D /var/lib/postgresql/14/main -c
config_file=/etc/postgresql/14/main/postgresql.conf
            9254 postgres: 14/main: checkpointer
            9255 postgres: 14/main: background writer
            9256 postgres: 14/main: walwriter
            9257 postgres: 14/main: autovacuum launcher
            9258 postgres: 14/main: stats collector
            9259 postgres: 14/main: logical replication launcher
 Oct 06 10:23:47 debian systemd[1]: Starting PostgreSQL Cluster 14-main…
 Oct 06 10:23:49 debian systemd[1]: Started PostgreSQL Cluster 14-main.

$ systemctl is-enabled postgresql
enabled
```

e. Во время установки автоматически создаётся пользователь postgres. Это суперадминистратор, который имеет полный доступ ко всему PostgreSQL.

4. Проверка соединения с PostgreSQL, настройка пользователя:
   a. Во время установки пользователь postgres создается автоматически. Этот пользователь имеет полный доступ superadmin ко всему экземпляру PostgreSQL.

```
sudo su - postgres
```

   b. (Необязательно) замените пароль пользователя на более надежный:

```
psql -c "alter user postgres with password 'NEW_PASSWORD'"
```

   c. Запускаем PostgreSQL с помощью команды:

```
$ psql
```

   d. Получить информацию о подключении, как показано ниже:

```
$ psql
psql (14.0 (Ubuntu 14.0-1.pgdg18.04+1))
Type "help" for help.

postgres=# \conninfo
You are connected to database "postgres" as user "postgres" via socket in "/var/run/postgresql"
at port "5432".
```

e. Убедиться, что сервис PostgreSQL запускается при загрузке системы, можно с помощью команд:

```
$ systemctl status postgresql.service
$ systemctl status postgresql@14-main.service
$ systemctl is-enabled postgresql
```

6.4. Настройка PostgreSQL

По умолчанию требуется развернуть кластер БД (ссылка и права доступа к инструкции по развертыванию кластера БД предоставляются по запросу заказчика)

Данный раздел следует использовать только в случае установки БД в режиме Standalone.

Установите postgresql-14 на вашу ОС.

PostgreSQL устанавливается и настраивается пропорционально количеству серверов.

После этого необходимо выполнить следующие настройки:

Необходим PostgreSQL версии 14 или выше.

1. Открыть конфигурационный файл ***postgresql.conf*** для редактирования. Файл находится здесь:

   ```
   /etc/postgresql/14/main/postgresql.conf
   ```

2. В файле выполнить следующее:

   - указать в файле значение параметра *timezone*, как показано ниже:

     ```
     timezone = 'UTC'
     ```

     Такое значение параметра необходимо при установке базы данных SMS.

   - изменить значение параметра *listen_addresses*, как показано ниже, и раскомментировать соответствующую строку:

```
listen_addresses = '*'          # what IP address(es) to listen on;
```

- для настройки автовакуума РЕКОМЕНДУЮТСЯ приведенные ниже значения:

```
autovacuum = on
#log_autovacuum_min_duration = 0
autovacuum_max_workers = 15
autovacuum_naptime = 1min
autovacuum_vacuum_threshold  = 120
autovacuum_analyze_threshold = 120
autovacuum_vacuum_scale_factor = 0.01
autovacuum_analyze_scale_factor = 0.05
```

- При данной конфигурации сервера:

CPU	Intel(R) Xeon(R) CPU E3-1220 v3 @ 3.10GHz
Memory	8 GB

рекомендуется использовать следующие параметры:

```
max_connections = 100
shared_buffers = 2GB
effective_cache_size = 6GB
work_mem = 20971kB
maintenance_work_mem = 512MB
wal_buffers = 16MB
default_statistics_target = 100
```

> Решающие параметры для запуска:
>
> *max_connections = 100*
>
> *shared_buffers = 2GB*
>
> Если объем памяти сервера меньше 8 ГБ, то значения указанных параметров должны быть меньше.

3. Открыть конфигурационный файл ***pg_hba.conf*** для редактирования. Файл находится здесь:

```
/etc/postgresql/14/main/pg_hba.conf
```

4. Необходимо, чтобы к postgres могли подсоединиться любые процессы с локальной машины, а также Docker-контейнеры (подсеть 172.17.0.0/16). Также необходимо указать настройки IPv6. Таким образом, файл может выглядеть следующим образом (пример):

```
# IPv4 local connections:
host        all                         all                         127.0.0.1
/32                                 trust
host        all                         all                         172.17.0.0
/16                                 trust

# IPv6 local connections:
host        all                         all                         ::1
/128                                                    trust
```

5. При работе CAS DREGUARD требуются поключения к базам данных. Необходимо настроить к ним доступ.
6. Перезапустить PostgreSQL:

```
sudo /etc/init.d/postgresql restart
```

6.5. Установка необходимых библиотек (для PostgreSQL)

Перед запуском скриптов для создания SMS_CAS_DB необходимо выполнить следующее:

1. Установить citus и pg_cron.
   Пример (для postgresql версии 14):

```
sudo apt-get -y install postgresql-14-citus postgresql-14-cron
```

2. Открыть файл ***postgresql.conf*** для редактирования. Файл находится здесь:

```
/etc/postgresql/14/main/postgresql.conf
```

3. Добавить установленные библиотеки в файл:

```
shared_preload_libraries = 'citus, pg_cron' # (change requires restart)
```

4. Добавить в файл название БД для pg_cron:

```
cron.database_name = 'smscas'
```

5. Убедиться, что в ***postgresql.conf*** следующее значение параметра *timezone*:

```
timezone = 'UTC'
```

6. Перезапустить PostgreSQL:

```
sudo /etc/init.d/postgresql restart
```

7. Установка Баз Данных

По умолчанию, БД будут созданы в табличном пространстве *pg_default* (табличное пространство по умолчанию для PostgreSQL).

7.1. Подготовка к установке

Порядок действий:

1. Распаковать архив ***rcas_scripts_X.X.X.zip*** , входящий в комплект поставки, в желаемую папку на разделе диска (например, */home*).
2. Предоставить пользователям права на чтение, изменение и запуск содержимого папки, созданной на предыдущем шаге:

```
sudo chmod -R +x /home/rcas_scripts
```

3. Сменить владельца данной папки на *postgres*:

```
sudo chown -R postgres /home/rcas_scripts
```

7.2. Установка БД

Развертывание всех баз данных, входящий в состав системы CAS, осуществляется из репозитория gitlab (с помощью CI/CD). См. Установка компонентов CAS DREGUARD.

⚠ **Обратите внимание!** При установке и развертывании баз данных с помощью CI/CD **названия баз данных не должны содержать "-" и "_" (дефисы и нижние подчеркивания)**. Поэтому здесь и далее соблюдайте это ограничение для всех разворачиваемых баз данных.

7.3. Наполнение CAS_DB

Получите данные, необходимые для наполнения CAS_DB и других баз, от поставщика ключей (Key Owner). Загрузите эти данные в базы CAS DREGUARD.

Описание этой процедуры выходит за рамки данного документа и приведено в отдельном документе (доступ строго ограничен).

7.4. Наполнение остальных БД

С помощью заполнения таблиц БД производится настройка:

- забора данных из SMS
- генерации OPKEY
- генерации и рассылки EMM

При установке в БД заносится начальное наполнение, с которым можно проводить тестовые запуски системы. В целом для тестового запуска необходимо добавить номер провайдера для всех заданий в таблице opkey. resources, количество ppvod ключей в таблице opkey.command_versions (оставьте 0 - неограниченное количество ключей) и выключить все неиспользуемые задания в таблице opkey.scheduler.

Однако при разворачивании боевой системы наполнение данной БД необходимо менять в соответствии с нуждами заказчика.

Общие принципы настройки БД описаны в соответствующем разделе документа "Руководстве администратора" (предоставляется по запросу заказчика).

7.5. Настройка автоматического запуска скрипта queue_health.sh

Данный скрипт лежит на сервере (в нашем примере располагается в */home/rcas_scripts/queue_health.sh*) и требуется для корректной работы *CAROUSEL DB*.

Порядок действий:

1. Создайте папку для запуска скрипта *queue_health*, например */usr/local/scripts*.

```
sudo mkdir /usr/local/scripts
```

2. Если название базы отличается от *crs*, то откройте скрипт на редактирование (в нашем примере скрипт находится в */home/rcas_scripts*), в строке '*-d crs*' замените *crs* на актуальное название базы данных.
3. Скопируйте скрипт (в нашем примере скрипт находится в */home/rcas_scripts*)

```
sudo cp /home/rcas_scripts/queue_health_cron.sh /usr/local/scripts/
```

4. Создайте файл с интрукцией для cron:

```
sudo nano /tmp/cron
```

5. В открывшемся редакторе введите текст и сохраните файл:

```
SHELL=/bin/bash
0-59 * * * * /usr/local/scripts/queue_health_cron.sh
```

   В конце файла обязательно должен быть переход на новую строку.
6. Добавьте указанное расписание в таблицу демона *cron*:

```
sudo crontab -u postgres /tmp/cron
```

7. Выполните команду:

```
sudo crontab -l -u postgres
```

   В результате должен отобразиться текст, добавленный из файла test.

8. Проверьте, что в выводе есть текст:

```
SHELL=/bin/bash
0-59 * * * * /usr/local/scripts/queue_health_cron.sh
```

9. Проверьте, что скрипт *queue_health_cron.sh* запущен. Если действия, указанные в этом шаге, не приводят к нужному результату, попробуйте перезагрузить машину (*sudo reboot*) и повторить действия шага.

> ⚠ Если скрипт не запускается, то:
>
> - проверьте, что установлены расширения dblink и pgstattuple. Если нет, установите расширения dblink и pgstattuple;
>
> - проверьте, что пользователь (по умолчанию - crs_admin) имеет права на выполнение функции dblink_connect_u. Если нет, выдайте соответствующие права.

Для проверки существует два варианта:

a. Через командную строку:
   i. Зайти в БД:

```
psql -d crs
```

   ii. Выполнить запрос для проверки того, что скрипт *queue_health.sh* запущен:

```
select query from pg_stat_activity where query like '%health%';
```

   iii. Убедиться, что в полученной выборке присутствует:

```
select carousel.queue_health();
```

   Если в выборке нет данной строки, попробуйте несколько раз в течение минуты выполнить указанный выше запрос, т.к. скрипт запускается раз в минуту.

b. PgAdmin или другие утилиты:
   i. Запустите PgAdmin
   ii. Подсоединитесь к созданной базе под пользователем *postgres*.
   iii. Откройте окно выполнения запросов и выполните следующий запрос:

```
select query from pg_stat_activity where query like '%health%';
```

   iv. Убедитесь, что в полученной выборке присутствует:

```
select.carousel.queue_health();
```

   Если в выборке нет данной строки, попробуйте несколько раз в течение минуты выполнить указанный выше запрос, т.к. скрипт запускается раз в минуту.

7.6. Настройка автоматического запуска процедуры flush_queue

Для корректного восстановления работы CAS DREGUARD после сбоев необходимо настроить автоматический запуск процедуры *flush_queue* при запуске/перезапуске Postgres.

1. Создайте новый файл в домашнем или другом удобном каталоге с названием check_state.sh со следущим содержимым:

check_state.sh

```
#!/bin/bash
state=`sudo -u postgres /usr/bin/psql -At -c 'select pg_is_in_recovery()'` 2>/dev/null
if [ "$state" = "t" ]; then
        echo "PostgeSQL server status is ReadOnly"
else
        /usr/bin/psql -U postgres -d crs -c 'select carousel.flush_queue()' 2>/dev/null
fi
exit 0
```

2. Откройте файл */lib/systemd/system/postgresql.service* для редактирования
3. После строки *ExecStart=/bin/true* добавьте следующую строку и сохраните изменения:

```
ExecStart=/bin/bash <path_file>/check_state.sh
```

4. Выполните команду:

```
sudo systemctl daemon-reload
```

Теперь при запуске/перезапуске Postgres будет автоматически запускаться процедура *flush_queue*.В случае если postgres работает в режиме read-only будет выводится в syslog "PostgeSQL server status is ReadOnly". Работу данной процедуры на боевом сервере можно отследить по изменению состояния Диспетчера (таблица *crs_queue_dispatch*), а также посредством выполнения команды:

```
sudo journalctl -xn
```

В случае если postgres работает в боевом режиме, в логе будет:

```
debian psql[15915]: flush_queue
debian psql[15915]: -------------
debian psql[15915]: 0
debian psql[15915]: (1 )
```

7.7. Настройка кэширования индивидуальных ключей

⚠ В конфигурационном файле EMMG есть параметр cache_individual_keys - флаг кэширования индивидуальных ключей.

Возможные значения:

- 0 - не кэшировать ключи: будет использоваться функция базы CAS DB individual_keys_get, которую необходимо установить отдельно при помощи скрипта individual_keys_get.sh (входит в состав rcas_scripts);
- 1 - кэширование включено: будет использоваться функция individual_keys_list, которая существует в CAS DB изначально.

По умолчанию cache_individual_keys: 0, поэтому необходимо выполнить скрипт individual_keys_get.sh

Процедура должны быть выполнена до начала работы (взаимодействия EMMG с CAS DB).

Поледовательность действий:

1. Перейти в папку с распакованной сборкой *rcas_scripts* (в нашем примере это */home/rcas_scripts*):

```
cd /home/rcas_scripts
```

2. Выполнить скрипт individual_keys_get.sh с параметрами.

ⓘ Команда запуска скрипта выглядит следующим образом:

bash individual_keys_get.sh [-help] <cas_version> <schema> [<host name> <port> <pg_user> <pg_password> <database>]

где:

- [-help] - вызов справки.
- <cas_version> - версия системы команд, в которой рабоатет CAS DB (для которой будет настраиваться кэширование ключей). Обязательный параметр.
- <schema> - схема базы, для которой осуществляется операция. Обязательный параметр.
- <host name> - имя хоста (hostname) CAS DB. Если БД находится на текущей машине, то параметр указывать не обязательно. Значение по умолчанию (задается в файле скрипта) - "localhost".
- <port> - номер порта CAS DB. Если БД находится на текущей машине, то параметр указывать не обязательно. Значение по умолчанию (задается в файле скрипта) - "5432".
- <pg_user> - имя пользователя базы данных. Если БД находится на текущей машине, то параметр указывать не обязательно. Значение по умолчанию (задается в файле скрипта) - "cas_dbadmin".
- <pg_password>- пароль для доступа к базе данных. Значение по умолчанию (задается в файле скрипта) - "cas_dbadmin".
- <database>- имя базы, для которой осуществляется операция. Значение по умолчанию (задается в файле скрипта) - "cas_db".

Пример:

```
bash individual_keys_get.sh 6 cas_26
```

3. Проверить лог файл *individual_keys_get.log* на отсутствие ошибок (находится в той же папке, что и скрипт).