ОТКРЫТАЯ МОБИЛЬНАЯ ПЛАТФОРМА

Прикладное программное обеспечение «Аврора Центр» релиз 5.7.0

РУКОВОДСТВО АДМИНИСТРАТОРА

Версия 1.4

Листов 281

2026

АННОТАЦИЯ

Настоящий документ является руководством администратора Прикладного программного обеспечения «Аврора Центр» (далее – ППО) релиз 5.7.0.

Настоящий документ содержит общую информацию о ППО, описание установки, обновления, удаления и резервного копирования ППО, описание управления сервисами и их настройками, а также информацию о конфигурационных файлах ППО.

СОДЕРЖАНИЕ

1. Общая информация 9
1.1. Назначение и состав ППО 9
1.1.1. Описание подсистем 12
1.1.2. Состав инфраструктурных компонентов ППО 18
1.2. Субъекты доступа и права на доступ к интерфейсам ППО 19
1.2.1. Субъекты доступа (роли) ППО 19
1.2.2. Права на доступ к интерфейсам ППО 20
1.3. Описание принципов безопасной работы средства 25
1.3.1. Общая информация 25
1.3.2. Компрометация паролей 25
1.3.3. Описание параметров (настроек) безопасности средства, доступных каждой роли пользователей, и их безопасные значения 26
1.4. Условия выполнения 26
1.4.1. Аппаратные характеристики 26
1.4.2. Программные характеристики 26
1.4.3. Требования к рабочим местам пользователей 31
1.4.4. Варианты конфигураций, для которых проводилось тестирование 31
1.5. Варианты поставки 33
1.6. Указания по эксплуатации 34
2. Архитектура ППО и варианты установки ППО 36
2.1. Описание компонентов 36
2.1.1. Сервер приложений 39
2.1.2. Сервер БД 41
2.1.3. Внешний балансировщик 41
2.1.4. Внешние службы 42
2.1.5. Сервер внешнего приложения 44
2.1.6. Перечень используемых внешних репозиториев 44
2.2. Внешние интерфейсы сервера приложений ППО 44
2.3. Варианты установки ППО 45
2.3.1. Требования к аппаратным характеристикам 45

2.3.2. Конфигурация все в одном (ППО и СУБД на одном сервере) 49
2.3.3. Конфигурация из одного сервера приложений и одного сервера БД 51
2.3.4. Кластерная конфигурация (поддержка до 10000 устройств) 54
2.3.5. Кластерная конфигурация с контент-серверами (поддержка до 100000 устройств) 55
2.3.6. Кластерная конфигурация с контент-сервером и раздельными серверами БД (поддержка до 500000 устройств) 57
2.3.7. Катастрофоустойчивая кластерная конфигурация с установкой серверов приложений и серверов БД в двух центрах обработки данных 59
3. Установка ППО 62
3.1. Общая информация 62
3.2. Порядок установки и настройки ОС на серверах приложений, серверах БД и контент-серверах 64
3.3. Порядок развертывания и настройки управляющей ЭВМ 69
3.4. Упрощенная настройка ППО с помощью интерактивного меню 73
3.5. Порядок настройки компонентов среды функционирования ППО и ППО 75
3.5.1. Настройка компонентов среды функционирования ППО и инфраструктурных компонентов ППО 75
3.5.2. Настройка ППО (подсистем ППО) 80
3.6. Порядок установки компонентов среды функционирования ППО и ППО 86
3.6.1. Установка компонентов среды функционирования ППО и инфраструктурных компонентов ППО 86
3.6.2. Установка ППО 90
3.6.3. Выполнение настройки подсистем ППО 91
3.6.4. Выполнение ограничений по применению 91
3.6.5. Проверка корректности установки и функционирования ППО 91
3.7. Адреса веб-консолей 91
3.8. Описание настройки подсистем ППО 92
3.8.1. Описание настройки ПСУ 92
3.8.2. Описание настройки ПУ 99
3.8.3. Описание настройки ПООС 103
3.8.4. Описание настройки CDN 107
3.9. Описание настройки файлового хранилища ППО 110

3.9.1. Настройка файловых хранилищ подсистем ППО 110
3.9.2. Настройка доступа нод сервера приложений ППО к файловому хранилищу 113
3.10. Дополнительные настройки ППО и среды функционирования ППО 116
3.10.1. Настройка взаимодействия сервера приложений ПУ с SMTP-сервером...116
3.10.2. Настройка взаимодействия сервера приложений ПУ с сервером удаленного подключения RustDesk 118
3.10.3. Настройка разделения трафика.....120
3.10.4. Настройка кэширования ответов сервисов 120
3.10.5. Действия по безопасной установке и настройке средства 122
3.10.6. Действия по смене аутентификационной информации (паролей, секретов, токенов, ключей).....128
3.10.7. Действия по реализации функций безопасности среды функционирования ППО и инфраструктурных компонентов ППО 129
3.10.8. Самостоятельная установка необходимых пакетов на серверы приложений, серверы БД и контент-серверы 133
3.10.9. Отключение служб SELinux и Firewalld 135
3.10.10. Требования к установке и настройке внешнего балансировщика (на примере Nginx).....136
3.10.11. Активация (разблокировка) учетной записи пользователя с помощью sql-запроса к БД 138
3.10.12. Действия после сброса устройств к заводским настройкам.....138
3.10.13. Порядок задания адресов (доменных имен) в инвентарном файле inventories/hosts.yml 139
3.10.14. Порядок настройки срока хранения событий безопасности 142
3.10.15. Порядок настройки срока хранения событий из журнала сообщений устройств 142
3.10.16. Порядок настройки ППО для его установки на различные окружения ...143
3.10.17. Удаление персональных данных из учетной записи пользователя, персональных данных контактного лица организации и персональных данных контактного лица проекта 144
3.10.18. Сброс пароля учетной записи.....147

3.10.19. Восстановление учетной записи пользователя тенанта в случае ее удаления 148
3.10.20. Настройка включения/отключения регистрации событий 149
3.10.21. Настройка брендирования ППО 150
3.10.22. Переключение трафика между ЦОД (failover/switchover) 151
3.10.23. Рекомендуемые действия при сильной рассинхронизации или зависании репликаци 152
3.10.24. Настройка адреса проверки подключения устройств к сети 153
3.10.25. Настройка подключения репозитория ОС Linux 155
3.10.26. Настройка защищенного TLS/SSL соединения с БД 156
3.10.27. Добавление корневого сертификата в доверенные для настройки защищенного TLS/SSL соединения 157
3.10.28. Настройка взаимодействия ПУ и ПСУ при использовании GOST TLS на внешнем балансировщике 158
3.10.29. Настройка передачи журнала аудита в SIEM-систему 160
3.10.30. Настройка интеграции с репозиторием «РТК-Феникс» 163
3.10.31. Установка и настройка PXE-сервера 164
3.10.32. Настройка подключения flatpak репозиториев 179
3.10.33. Настройка интеграции с S3 хранилищем 182
3.10.34. Порядок получения метрик ППО 183
3.10.35. Настройка удаленного пробуждения компьютера в КПСД 185
3.10.36. Настройка проверки срока жизни управляемых переменных 186
3.10.37. Настройка интеграции с внешней системой Platform V IAM SE 187
3.10.38. Настройка проверки для очистки файлов билдов и зависимостей из хранилища 191
3.10.39. Настройки очистки данных при удалении устройств 192
3.11. Проверка корректности установки и функционирования ППО 192
3.11.1. Общие сведения 192
3.11.2. Описание параметров диагностического отчета 193
3.12. Самостоятельная установка и настройка СУБД Postgres Pro и СУБД PostgreSQL 14/15/16/17 202
4. Управление компонентами среды функционирования ППО, инфраструктурными компонентами ППО, сервисами, настройками сервисов и подсистем 209

4.1. Управление компонентами среды функционирования ППО и инфраструктурными компонентами ППО....209
4.2. Управление сервисами ППО....212
4.3. Управление настройками сервисов и подсистем ППО....217
4.3.1. Способ 1 (рекомендуемый)....217
4.3.2. Способ 2....217
5. Резервное копирование....218
5.1. Резервное копирование после установки (обновления) ППО....218
5.2. Периодическое резервное копирование и резервное копирование перед установкой обновлений....218
6. Обновление ППО и ОС Аврора....219
6.1. Порядок обновления....219
6.2. Обновление сервера приложений ППО....221
6.3. Обновление и установка ППО при использовании единого docker-образа (All-in-One)....222
6.4. Обновление ОС Аврора с помощью ПУ....225
7. Удаление ППО....226
8. Варианты установки ПСУ....227
8.1. Установка ПСУ на один сервер (хост) с другими подсистемами ППО....227
8.2. Установка ПСУ на отдельный сервер (хост)....227
8.3. Отдельная установка ПСУ (установка ПБ и ПСУ)....227
9. Варианты установки СУБД....228
9.1. Некластерная (standalone) установка СУБД....228
9.2. Установка СУБД в кластерной конфигурации....228
10. Установка ППО в Kubernetes....232
10.1. Порядок развертывания и настройки сервера приложений....232
10.2. Порядок установки ППО в Kubernetes....235
10.3. Порядок удаления ППО из Kubernetes....243
11. Импорт отчетов и настройка автоматической рассылки в Grafana....244
11.1. Порядок импорта отчетов в Grafana....244
11.2. Установка и настройка плагина для автоматической рассылки отчетов из Grafana на почту....248
11.3. Импорт дашборда в Grafana с использованием Prometheus....259

12. Конфигурационные файлы сценариев установки среды функционирования ППО и инфраструктурных компонентов ППО 261
12.1. Конфигурационные файлы сценариев установки среды функционирования ППО и инфраструктурных компонентов ППО 261
12.1.1. Инвентарный файл inventories/hosts.yml 261
12.1.2. Настройки сценариев установки среды функционирования ППО и инфраструктурных компонентов ППО в конфигурационных файлах config/vars/_vars.yml и config/subsystems/<название подсистемы>/vars/_vars.yml 263
12.1.3. Настройки паролей и секретов компонентов среды функционирования ППО и инфраструктурных компонентов ППО в конфигурационных файлах config/secret.yml и config/subsystems/<название подсистемы>/secret.yml 263
13. Конфигурационные файлы ППО (сценариев установки ППО) 264
13.1. Общая информация о конфигурационных файлах ППО 264
13.2. Общая информация о конфигурационных файлах сценариев установки ППО 266
13.2.1. Инвентарный файл inventories/hosts.yml 267
13.2.2. Общий конфигурационный файл сценариев установки config/vars/_vars.yml 267
13.2.3. Конфигурационные файлы сценариев установки для подсистем ППО (файлы: config/subsystems/<название подсистемы>/vars/_vars.yml) 268
13.2.4. Шаблоны конфигурационных файлов ППО и подсистем ППО 268
13.2.5. Конфигурационный файл с паролями и токенами компонентов среды функционирования ППО и инфраструктурных компонентов ППО config/secret.yml 269
13.2.6. Конфигурационные файлы подсистем ППО 269
13.2.7. Конфигурационные файлы сервисов ППО 269
13.2.8. Конфигурационные файлы окружений 270
13.2.9. Порядок работы с конфигурационными файлами сценариев установки ППО 271
14. Гарантийные обязательства 274
Перечень терминов и сокращений 276

1. ОБЩАЯ ИНФОРМАЦИЯ

1.1. Назначение и состав ППО

ППО является прикладным программным обеспечением, предназначенным для:

– управления устройствами[1], а также приложениями[2] и их жизненным циклом;

– отправки push-уведомлений на устройства;

– обновления операционной системы (ОС) Аврора и ОС семейства Linux путем получения из доверенного хранилища пакетов с изменениями ОС (образа ОС) и их установки. При этом указанные процессы выполняются штатными средствами самой ОС, а ППО участвует лишь в их инициализации в ОС и не гарантирует их успешного завершения.

ППО состоит из следующих подсистем:

– подсистема безопасности (ПБ);

– подсистема «Маркет» (ПМ);

– подсистема Платформа управления (ПУ);

– подсистема управления тенантами (ПУТ);

– подсистема Сервис уведомлений (ПСУ);

– подсистема обновления ОС (ПООС);

– подсистема доставки контента (CDN).

ПРИМЕЧАНИЕ. ППО состоит также из инфраструктурных компонентов, приведенных в п. 1.1.2.

[1] Определение термина «Устройство» приведено в таблице (Таблица 36).
[2] Определение термина «Приложение» приведено в таблице (Таблица 36).

Взаимодействие между подсистемами и компонентами подсистем осуществляется с использованием протокола HTTP стандарт RFC 2616, при этом обмен данными осуществляется в формате RFC 8259 (JSON).

Для получения push-уведомлений на устройства используется push-демон. Push-демон, в свою очередь, взаимодействует с ПСУ по защищенному протоколу TLS (RFC 5246, RFC 8446) с протоколом TCP (RFC 793) на транспортном уровне.

В качестве сервера базы данных (БД) используется сервер с одной из следующих установленных систем управления базами данных (СУБД) Postgres Pro, Platform V Pangolin или PostgreSQL, в которой хранятся данные ППО, для чего при развертывании создается специальная БД.

Для хранения информации о сессиях используется СУБД Valkey или Platform V Radish.

ПРИМЕЧАНИЕ. Подсистемы, входящие в состав ППО, выполняют логирование в системный журнал ОС (`systemd-journald`) следующей информации:

- информационных сообщений;
- сообщений об ошибках;
- предупреждений;
- отладочной информации.

В ППО реализована возможность интеграции с внешней системой управления доступом к информационным ресурсам (СУДИР), построенной с использованием сертифицированного программного обеспечения (ПО) «Platform V IAM SE»[3].

Platform V IAM SE (Identity & Access Management) – набор инструментов для управления доступом к информационным ресурсам, включающий следующие компоненты:

- IAM Keycloak – провайдер идентификации, представляет собой хранилище учетных записей и информации о пользователях;

[3] Сертификат соответствия ФСТЭК России № 4999, действителен до 26 ноября 2030 г.

– IAM Proxy – реверсивный прокси-сервер, обеспечивающий аутентифицированный доступ к ресурсам.

ВНИМАНИЕ! В случае интеграции с СУДИР функции ПБ полностью или частично передаются СУДИР. При интеграции ППО с СУДИР Консоль входа пользователей и Консоль администратора ПБ становятся недоступны. В таком случае аутентификация пользователей в ППО и управление учетными записями выполняются через компонент IAM Keycloak ПО «Platform V IAM SE».

Описание интерфейсов подсистем, входящих в состав ППО, приведено в следующих документах:

– «Руководство пользователя. Часть 1. Подсистема безопасности»;

– «Руководство пользователя. Часть 2. Подсистема «Маркет»;

– «Руководство пользователя. Часть 3. Подсистема Платформа управления»;

– «Руководство пользователя. Часть 4. Подсистема управления тенантами»;

– «Руководство пользователя. Часть 5. Подсистема Сервис уведомлений».

Описание работы приложений приведено в документах:

– «Руководство пользователя. Часть 6. Приложение «Аврора Маркет» для операционной системы Аврора»;

– «Руководство пользователя. Часть 7. Приложение «Аврора Центр» для операционной системы Аврора»;

– «Руководство пользователя. Часть 8. Приложение «Аврора Маркет» для операционной системы Android»;

– «Руководство пользователя. Часть 9. Приложение «Аврора Центр» для операционной системы Android»;

– «Руководство пользователя. Часть 10. Приложение «Аврора Маркет» для операционных систем семейства Linux»;

– «Руководство пользователя. Часть 11. Приложение «Аврора Центр» для операционных систем семейства Linux».

1.1.1. Описание подсистем

1.1.1.1. Подсистема безопасности

ПРИМЕЧАНИЕ. Более подробная информация о ПБ приведена в документе «Руководство пользователя. Часть 1. Подсистема безопасности».

ПБ предназначена для реализации следующих функций безопасности ППО:

– идентификации и аутентификации пользователей, устройств и внешних сервисов[4];

– управления идентификаторами пользователей, устройств и внешних сервисов;

– управления средствами аутентификации;

– управления учетными записями пользователей и устройств;

– управления доступом субъектов доступа к объектам доступа;

– регистрации событий безопасности;

– предоставления пользователям доступа к интерфейсу ПБ.

ПБ состоит из следующих компонентов:

– Консоль входа пользователей;

– Консоль администратора ПБ;

– Сервер приложений ПБ.

Консоль входа пользователей позволяет пользователям ППО осуществлять ввод идентификационной и аутентификационной информации.

Консоль администратора ПБ позволяет управлять учетными записями пользователей и техническими учетными записями, а также работать с журналом регистрации событий.

[4] Под внешними сервисами подразумеваются любые системы, взаимодействующие с ППО через его API.

Сервер приложений ПБ представляет собой совокупность веб-приложений, реализующих функции безопасности, а также позволяющих хранить в БД и предоставлять пользователям ППО доступ к данным об учетных записях и журналу регистрации событий.

1.1.1.2. Подсистема «Маркет»

ПРИМЕЧАНИЕ. Более подробная информация о ПМ приведена в следующих документах:

– «Руководство пользователя. Часть 2. Подсистема «Маркет»;

– «Руководство пользователя. Часть 6. Приложение «Аврора Маркет» для операционной системы Аврора»;

– «Руководство пользователя. Часть 8. Приложение «Аврора Маркет» для операционной системы Android»;

– «Руководство пользователя. Часть 10. Приложение «Аврора Маркет» для операционных систем семейства Linux».

ПМ предназначена для обеспечения:

– управления жизненным циклом приложений (загрузка, согласование, удаление и публикация);

– управления дистрибуцией опубликованных приложений (скачивание, установка, обновление и удаление);

– предоставления пользователям доступа к интерфейсу ПМ.

ПМ состоит из следующих компонентов:

– Консоль администратора ПМ;

– Консоль разработчика ПМ;

– Приложение «Аврора Маркет»;

– Сервер приложений ПМ.

Консоль администратора ПМ позволяет осуществлять взаимодействие Администратора Аврора Маркета с ПМ в части работы с приложениями.

Консоль разработчика ПМ позволяет добавлять новые и обновлять ранее загруженные приложения, а также получать доступ к хранимой информации о приложениях.

Приложение «Аврора Маркет» служит для отображения данных о приложениях, а также для их загрузки, установки, обновления и удаления.

Сервер приложений ПМ представляет собой совокупность веб-приложений, позволяющих хранить в БД и предоставлять пользователям ППО информацию о приложениях. При этом сами приложения, их иконки и скриншоты хранятся в файловом хранилище.

1.1.1.3. Подсистема Платформа управления

ПРИМЕЧАНИЕ. Более подробная информация о ПУ приведена в следующих документах:

– «Руководство пользователя. Часть 3. Подсистема Платформа управления»;

– «Руководство пользователя. Часть 7. Приложение «Аврора Центр» для операционной системы Аврора»;

– «Руководство пользователя. Часть 9. Приложение «Аврора Центр» для операционной системы Android»;

– «Руководство пользователя. Часть 11. Приложение «Аврора Центр» для операционных систем семейства Linux».

ПУ предназначена для обеспечения:

– управления отдельными устройствами (оперативное управление) и группами устройств;

– управления политиками, офлайн-сценариями;

– управления записями об устройствах и пользователях устройств;

– управления приложениями на устройствах;

– контроля состояния устройств;

– контроля применения политик на устройствах;

– мониторинга событий и предоставления отчетности;

– мониторинга геолокации устройств;

– загрузки файлов и папок в ППО с локального компьютера или из подключенных git-репозиториев для последующего распространения на устройства;

– согласования загруженных файлов или папок перед распространением на устройства;

– предоставления пользователям доступа к интерфейсу ПУ.

ПУ состоит из следующих компонентов:

– Консоль администратора ПУ;

– Приложение «Аврора Центр»;

– Сервер приложений ПУ.

Консоль администратора ПУ позволяет осуществлять взаимодействие Администратора Платформы управления с ПУ.

Приложение «Аврора Центр» выполняется на устройствах, функционирующих под управлением ОС, и позволяет осуществлять взаимодействие ПУ с устройством, а также в зависимости от управляющего сообщения или назначенного офлайн-сценария, полученного от Сервера приложений ПУ, имеет возможность управлять различными функциями устройства.

Сервер приложений ПУ представляет собой совокупность веб-приложений, позволяющих хранить в БД и предоставлять пользователям ППО данные о настройках и конфигурации ОС, а также формировать управляющие сообщения и офлайн-сценарии для приложения «Аврора Центр».

1.1.1.4. Подсистема управления тенантами

ПРИМЕЧАНИЕ. Более подробная информация о ПУТ приведена в документе «Руководство пользователя. Часть 4. Подсистема управления тенантами».

ПУТ предназначена для обеспечения:

– управления жизненным циклом тенантов (создание, редактирование и удаление тенантов);

– управления справочником организаций, использующих тенанты;

– управления информацией о контактных лицах организаций, использующих тенанты;

ПУТ состоит из следующих компонентов:

– Консоль администратора ПУТ;

– Сервер приложений ПУТ.

Консоль администратора ПУТ позволяет осуществлять взаимодействие Администратора тенантов с ПУТ.

Сервер приложений ПУТ представляет собой совокупность веб-приложений, позволяющих хранить в БД и предоставлять пользователям ППО данные о тенантах, а также осуществлять управление тенантами.

1.1.1.5. Подсистема Сервис уведомлений

ПРИМЕЧАНИЕ. Более подробная информация о ПСУ приведена в документе «Руководство пользователя. Часть 5. Подсистема Сервис уведомлений».

ПСУ предназначена для обеспечения:

– доставки push-уведомлений до устройств под управлением ОС Аврора и ОС Android;

– управления жизненным циклом проектов (добавление, настройка и удаление);

– предоставления пользователям доступа к интерфейсу ПСУ.

ПСУ состоит из следующих компонентов:

– Консоль администратора ПСУ;

– Сервер приложений ПСУ.

Консоль администратора ПСУ позволяет осуществлять взаимодействие Администратора Сервиса уведомлений с ПСУ в части управления жизненным циклом проектов. При этом проекты содержат следующую информацию:

– настройки взаимодействия ПСУ и сервера внешнего приложения;

– информацию о внешних приложениях, push-уведомления от которых передаются на устройства;

– информацию о контактных лицах.

Сервер приложений ПСУ представляет собой совокупность веб-приложений, позволяющих хранить в БД и предоставлять субъектам доступа ППО информацию о проектах, а также реализует функционал доставки push-уведомлений до устройств посредством tcp-сервера.

1.1.1.6. Подсистема обновления ОС

ПООС предназначена для обеспечения:

– предоставления информации о пакетах ОС;

– управления дистрибуцией пакетов ОС.

ПООС состоит из следующего компонента:

– Сервер приложений ПООС.

Сервер приложений ПООС представляет собой совокупность веб-приложений, позволяющих хранить в БД и предоставлять информацию и адреса хранения пакетов загрузочного модуля ОС.

Для хранения и дистрибуции пакетов ОС применяется файловый сервер, развернутый с использованием Nginx.

1.1.1.7. Подсистема доставки контента

CDN является опциональной подсистемой ППО и предназначена для оптимизации доставки контента ППО (установочные файлы приложений, значки, скриншоты, пакеты обновления ОС) путем их размещения (кеширования) на контент-

серверах таким образом, чтобы время ожидания для пользователя было минимальным.

CDN состоит из следующего компонента:

– контент-сервера (контент-серверов).

Контент-сервер представляет собой веб-приложение, позволяющее кешировать в файловом хранилище контент ППО и предоставлять к нему доступ приложениям «Аврора Центр» и «Аврора Маркет», а также ОС, на которую выполняется доставка контента.

1.1.2. Состав инфраструктурных компонентов ППО

К инфраструктурным компонентам относятся компоненты, которые не реализуют целевой функционал ППО и выполняют сервисные функции: управление сервисами, транспортировка запросов, хранение информации, синхронизация файлов в распределенном файловом хранилище, запуск задач по расписанию, партиционирование таблиц в БД.

ППО состоит из следующих инфраструктурных компонентов:

– внутренний балансировщик;

– система обнаружения сервисов Consul;

– средство управления конфигурациями сервисов Consul Template;

– сервис гарантированной доставки сообщений Redpanda;

– СУБД Valkey (fork СУБД Redis);

– расширение PG Partition Manager (pg_partman) для СУБД PostgreSQL и Postgres Pro;

– планировщик задач pg_cron для СУБД PostgreSQL и Postgres Pro;

– приложение для синхронизации файлов Syncthing;

– агрегатор (коллектор) для сбора метрик OpenTelemetry Collector.

Более подробное описание инфраструктурных компонентов приведено в подразделе 2.1.

1.2. Субъекты доступа и права на доступ к интерфейсам ППО

1.2.1. Субъекты доступа (роли) ППО

Субъектами доступа могут являться:

– пользователи ППО;

– процессы без участия пользователей (устройства и внешние сервисы[5]).

Субъектам доступа (кроме внешних сервисов) может быть назначена одна или несколько ролей[6], позволяющих выполнять следующие действия:

– Администратор учетных записей – управлять учетными записями пользователей (наличие роли обязательно в ППО);

– Администратор Аврора Маркета – управлять ПМ через интерфейс ППО;

– Администратор Платформы управления – управлять ПУ через интерфейс ППО;

– Администратор устройств Платформы управления – управлять ПУ через интерфейс ППО без доступа к изменениям критических настроек;

– Администратор тенантов – управлять ПУТ через интерфейс ППО;

– Администратор Сервиса уведомлений – управлять жизненным циклом проектов;

– Специалист технического обслуживания устройств Платформы управления – управлять ПУ через интерфейс ППО без доступа управлением парком устройств;

– Специалист технической поддержки Платформы управления – осуществлять удаленную поддержку пользователей;

– Оператор аудита – работать с журналом регистрации событий;

[5] Под внешними сервисами подразумеваются любые системы (процессы в системе без участия пользователей), взаимодействующие с ППО через его API. В отношении доступа внешних сервисов к ППО реализован дискреционный метод управления доступом.

[6] Более подробное описание некоторых ролей, приведено в приложении документа «Руководство пользователя. Часть 1. Подсистема безопасности».

– Разработчик – добавлять новые, обновлять ранее загруженные приложения и получать информацию о них;

– Редактор приложений – обновлять и получать информацию о ранее загруженных приложениях;

– Пользователь Аврора Маркета – загружать приложения и получать информацию о них;

– Мобильное приложение (процесс без участия пользователей) – получать push-уведомления;

– Приложение «Аврора Центр» (процесс без участия пользователей) – назначается учетным записям приложения «Аврора Центр»;

– Сервер внешнего приложения (процесс на сервере без участия или с участием пользователей) – назначается серверам внешних приложений, отправляющих push-уведомления на Сервер приложений ПСУ, для их последующей передачи в соответствующее мобильное приложение.

1.2.2. Права на доступ к интерфейсам ППО

Права на доступ к соответствующим разделам интерфейса ППО приведены в таблице (Таблица 1).

Таблица 1

Интерфейс ППО		Права на доступ		
Раздел	Подраздел	Подсистема	Консоль	Субъект доступа
Мультитенант	Тенанты	ПУТ	Консоль администратора ПУТ	Администратор тенантов
	Организации	ПУТ	Консоль администратора ПУТ	Администратор тенантов
Мониторинг	Индикаторы	ПУ	Консоль администратора ПУ	– Администратор Платформы управления; – Администратор устройств Платформы управления; – Специалист технического обслуживания устройств; – Специалист технической поддержки Платформы управления
	Аудит	ПБ	Консоль администратора ПБ	Оператор аудита
Управление	Устройства	ПУ	Консоль администратора ПУ	– Администратор Платформы управления; – Администратор устройств Платформы управления; – Специалист технического обслуживания устройств; – Специалист технической поддержки Платформы управления

Интерфейс ППО		Права на доступ		
Раздел	Подраздел	Подсистема	Консоль	Субъект доступа
	Пользователи	ПУ	Консоль администратора ПУ	– Администратор Платформы управления; – Администратор устройств Платформы управления; – Специалист технического обслуживания устройств; – Специалист технической поддержки Платформы управления
	Политики	ПУ	Консоль администратора ПУ	– Администратор Платформы управления; – Администратор устройств Платформы управления; – Специалист технического обслуживания устройств; – Специалист технической поддержки Платформы управления
	Сценарии	ПУ	Консоль администратора ПУ	– Администратор Платформы управления; – Администратор устройств Платформы управления; – Специалист технического обслуживания устройств; – Специалист технической поддержки Платформы управления

Интерфейс ППО		Права на доступ		
Раздел	Подраздел	Подсистема	Консоль	Субъект доступа
	Файлы	ПУ	Консоль администратора ПУ	– Администратор Платформы управления; – Администратор устройств Платформы управления; – Специалист технического обслуживания устройств; – Специалист технической поддержки Платформы управления
	Приложения	ПМ	Консоль администратора ПМ	Администратор Аврора Маркета
	Витрины	ПМ	Консоль администратора ПМ	Администратор Аврора Маркета
	Связки ключей	ПМ	Консоль администратора ПМ	Администратор Аврора Маркета
Администрирование	Учетные записи	ПБ	Консоль администратора ПБ	Администратор учетных записей
	Настройки	ПУ	Консоль администратора ПУ	– Администратор Платформы управления; – Администратор устройств Платформы управления; – Специалист технического обслуживания устройств; – Специалист технической поддержки Платформы управления

Интерфейс ППО		Права на доступ		
Раздел	Подраздел	Подсистема	Консоль	Субъект доступа
	Орг. структура	ПУ	Консоль администратора ПУ	– Администратор Платформы управления; – Администратор устройств Платформы управления; – Специалист технического обслуживания устройств; – Специалист технической поддержки Платформы управления
	Версии ОС	ПМ	Консоль администратора ПМ	Администратор Аврора Маркета
Консоль разработчика ПМ		ПМ	Консоль разработчика ПМ	Разработчик, Редактор приложений
Проекты		ПСУ	Консоль администратора ПСУ	Администратор Сервиса уведомлений
Приложение «Аврора Маркет»		ПМ	Пользователь Аврора Маркета	
Приложение «Аврора Центр»		ПУ	Процесс приложения «Аврора Центр»	

1.3. Описание принципов безопасной работы средства

1.3.1. Общая информация

ППО реализует следующие функции безопасности:

– идентификация и аутентификация субъектов доступа и объектов доступа;

– управление доступом субъектов доступа к объектам доступа;

– регистрация событий безопасности.

При использовании ППО необходимо выполнение следующих мер по защите информации от несанкционированного доступа (НСД):

– соблюдение парольной политики;

– соблюдение требования, согласно которому пароль не должен включать в себя легко вычисляемые сочетания символов;

– отсутствие у пользователя права передачи личного пароля третьим лицам;

– обязанность пользователя при вводе пароля исключить возможность его перехвата третьими лицами и техническими средствами.

При эксплуатации ППО запрещается:

– оставлять без контроля незаблокированные программные средства и/или ППО;

– разглашать пароли, выводить пароли на дисплей, принтер или иные средства отображения информации.

1.3.2. Компрометация паролей

Под компрометацией паролей необходимо понимать следующее:

– физическую утрату носителя с парольной информацией;

– передачу идентификационной информации по открытым каналам связи;

– перехват пароля при распределении идентификаторов;

– сознательную передачу информации третьим лицам.

ПРИМЕЧАНИЕ. При компрометации пароля пользователь обязан незамедлительно оповестить Администратора учетных записей.

1.3.3. Описание параметров (настроек) безопасности средства, доступных каждой роли пользователей, и их безопасные значения

Настройки параметров безопасности ППО доступны только пользователям с ролью Администратор учетных записей и заключаются в возможности управления ролями пользователей ППО.

Пользователям ППО должны назначаться минимальные права и привилегии, необходимые для выполнения ими своих должностных обязанностей (функций).

1.4. Условия выполнения

Для функционирования ППО необходимы описанные в настоящем подразделе программно-технические средства.

1.4.1. Аппаратные характеристики

Аппаратные характеристики приведены в п. 2.3.1.

1.4.2. Программные характеристики

В таблице (Таблица 2) приведены программные характеристики электронно-вычислительной машины (ЭВМ), на которой располагается сервер приложений ППО.

Таблица 2

Параметр	Значение	Информация о лицензии
Операционная система	Ubuntu версии 22.04	Free software, plus some proprietary device drivers
	Ubuntu версии 24.04	Free software, plus some proprietary device drivers
	Debian версии 12	DFSG-compatible licenses, plus proprietary firmware files
	Альт 8 СП[7] релиз 10	Коммерческая

[7] Сертификат соответствия ФСТЭК России № 3866, действителен до 10 августа 2028 г.

Параметр	Значение	Информация о лицензии
	Альт Сервер 10	
	РЕД ОС 7.3[8] (сертифицированный)	
	РЕД ОС 7.3	
	РЕД ОС 8.0	
	Astra Linux Special Edition 1.7[9] (Орел, Воронеж)	
	Astra Linux Special Edition 1.8[10] (Орел, Воронеж, Смоленск)	
	Platform V SberLinux OS Server 9.2 (fstec)[11]	
	Platform V SberLinux OS Server 10.2.1	
Универсальный базовый образ операционной системы	Ubuntu версии 22.04	Free software, plus some proprietary device drivers
	Platform V SberLinux OS 9	Коммерческая
	РЕД ОС версии 7.3[12] (сертифицированный)	
	Альт 8 СП[13] релиз 10	
	Astra Linux Special Edition 1.8[14]	
Прикладное программное обеспечение	ППО «Аврора Центр»	Коммерческая

[8] Сертификат соответствия ФСТЭК России № 4060, действителен до 12 января 2024 г. (окончание срока технической поддержки 31.12.2030 г.).

[9] Сертификат соответствия ФСТЭК России № 2557, действителен до 27 января 2026 г. (окончание срока технической поддержки 31.12.2050 г.).

[10] Сертификат соответствия ФСТЭК России № 2557, действителен до 27 января 2026 г. (окончание срока технической поддержки 31.12.2050 г.).

[11] Сертификат соответствия ФСТЭК России № 4884, действителен до 04 декабря 2029 г.

[12] Сертификат соответствия ФСТЭК России № 4060, действителен до 12 января 2024 г. (окончание срока технической поддержки 31.12.2030 г.).

[13] Сертификат соответствия ФСТЭК России № 3866, действителен до 10 августа 2028 г.

[14] Сертификат соответствия ФСТЭК России № 2557, действителен до 27 января 2026 г. (окончание срока технической поддержки 31.12.2050 г.).

Параметр	Значение	Информация о лицензии
Инфраструктурные компоненты ППО		
Балансировщик сервисов	Nginx Web Server версии 1.31.6 или выше	2-clause BSD-like license
	Platform V SynGX	Коммерческая
Система обнаружения сервисов	Consul версии 2.0.4 или выше	Mozilla Public License, version 2.0
Средство управления конфигурациями сервисов	Consul Template версии 0.43.0 или выше	Mozilla Public License, version 2.0
Сервис гарантированной доставки сообщений	Redpanda версии 26.2.2 или выше	Redpanda Business Source License 1.1 (BSL 1.1)
	Platform V Corax	Коммерческая
Приложение для синхронизации файлов	Syncthing версии 2.1.3 или выше	Mozilla Public License, version 2.0
Система управления контейнерами	Platform V DropApp (fstec)[15]	Коммерческая
	Platform V DropApp	

ПРИМЕЧАНИЕ. ОС должна быть установлена в минимальной конфигурации без графического интерфейса.

В таблице (Таблица 3) приведены программные характеристики ЭВМ, на которой располагается сервер БД.

Таблица 3

Параметр	Значение	Информация о лицензии
Операционная система	Ubuntu версии 22.04	Free software, plus some proprietary device drivers
	Ubuntu версии 24.04	Free software, plus some proprietary device drivers
	Debian версии 12	DFSG-compatible licenses, plus proprietary firmware files
	Альт 8 СП релиз 10	Коммерческая
	Альт Сервер 10	
	РЕД ОС 7.3 (сертифицированный)	
	РЕД ОС 7.3	

[15] Сертификат соответствия ФСТЭК России № 4883, действителен до 04 декабря 2029 г.

Параметр	Значение	Информация о лицензии
	РЕД ОС 8.0	
	Astra Linux Special Edition 1.7 (Орел, Воронеж)	
	Astra Linux Special Edition 1.8 (Орел, Воронеж, Смоленск)	
	Platform V SberLinux OS Server 9.2 (fstec)	
	Platform V SberLinux OS Server 10.2.1	
Универсальный базовый образ операционной системы	Ubuntu версии 22.04	Free software, plus some proprietary device drivers
	Platform V SberLinux OS 9	Коммерческая
	РЕД ОС версии 7.3[16] (сертифицированный)	
	Альт 8 СП[17] релиз 10	
	Astra Linux Special Edition 1.8[18]	
СУБД	Postgres Pro Certified 14.24.1 или выше	Коммерческая
	Postgres Pro Certified 15.19.1 или выше	
	Postgres Pro Certified 16.15.1 или выше	
	Postgres Pro Certified 17.11.1 или выше	
	Postgres Pro Standard 14.23.1 или выше	
	Postgres Pro Standard 15.18.1 или выше	
	Postgres Pro Standard 16.14.1 или выше	
	Postgres Pro Standard 17.10.1 или выше	

[16] Сертификат соответствия ФСТЭК России № 4060, действителен до 12 января 2024 г. (окончание срока технической поддержки 31.12.2030 г.).

[17] Сертификат соответствия ФСТЭК России № 3866, действителен до 10 августа 2028 г.

[18] Сертификат соответствия ФСТЭК России № 2557, действителен до 27 января 2026 г. (окончание срока технической поддержки 31.12.2050 г.).

Параметр	Значение	Информация о лицензии
	Platform V Pangolin DB 6.4.2-cve2-fstec[19] или выше	
	Platform V Pangolin DB 6	
	Platform V Pangolin DB 7	
	PostgreSQL 14.24 или выше	PostgreSQL License
	PostgreSQL 15.19 или выше	
	PostgreSQL 16.15 или выше	
	PostgreSQL 17.11 или выше	
	PostgreSQL 18.5 или выше	
Инфраструктурные компоненты ППО		
СУБД для хранения сессий	Valkey версии 9.1.2 или выше	BSD-3-Clause License
	Platform V Radish	Коммерческая
Расширение СУБД PostgreSQL для партиционирования таблиц БД	PG Partition Manager (`pg_partman`) версии 5.1.0 или выше	PostgreSQL License
Планировщик задач для PostgreSQL	`pg_cron` версии 1.6.2 или выше	PostgreSQL License
Сервис для управления кластером Postgresql	Patroni версии 3.3.0 или выше	The MIT License (MIT)
Сервис для балансировки нагрузки и обеспечения отказоустойчивости	Keepalived	GNU General Public License, version 2
Система управления контейнерами	Platform V DropApp (fstec)	Коммерческая
	Platform V DropApp	

ПРИМЕЧАНИЕ. ОС должна быть установлена в минимальной конфигурации без графического интерфейса.

В таблице (Таблица 4) приведены программные характеристики устройств.

[19] Сертификат соответствия ФСТЭК России № 4704, действителен до 22 августа 2028 г.

Таблица 4

Параметр	Значение
Операционная система	ОС Аврора, ОС Android, ОС семейства Linux
Прикладное программное обеспечение	– приложение «Аврора Центр»; – приложение «Аврора Маркет»

1.4.3. Требования к рабочим местам пользователей

ПРИМЕЧАНИЕ. Для работы пользователей с интерфейсом ППО необходимо выполнение следующих условий:

– веб-браузер должен поддерживать следующие технологии: TLS, CSS3, HTML5, ECMAScript 5 и Cookie. Рекомендуется использовать веб-браузер Chrome версии 90 или выше;

– разрешение экрана монитора не менее 1280x960 px.

1.4.4. Варианты конфигураций, для которых проводилось тестирование

Варианты конфигурации среды функционирования, в которых проводилось тестирование ППО, приведены в таблице (Таблица 5).

Таблица 5

ОС	СУБД
Альт Сервер 10.4	PostgreSQL 14.24
Альт Сервер 10.4	PostgreSQL 15.19
Альт Сервер 10.4	PostgreSQL 16.15
Альт Сервер 10.4	PostgreSQL 17.11
Альт 8 СП релиз 10.2.2	PostgreSQL 15.19
Альт 8 СП релиз 10.2.2	PostgreSQL 16.15
Альт 8 СП релиз 10.2.2	Postgres Pro Standard 16.15.1
Альт 8 СП релиз 10.2.2	Postgres Pro Standard 17.11.1
Альт 8 СП релиз 10.2.2	Postgres Pro Certified (версия ядра postgres: 15.18.1)
РЕД ОС 7.3 (сертифицированный)	Postgres Pro Certified (версия ядра postgres: 14.23.1)
РЕД ОС 7.3 (сертифицированный)	Postgres Pro Certified (версия ядра postgres: 15.18.1)
РЕД ОС 7.3.7	PostgreSQL 14.23
РЕД ОС 7.3.7	PostgreSQL 15.18

ОС	СУБД
РЕД ОС 7.3.7	PostgreSQL 16.14
РЕД ОС 7.3.7	PostgreSQL 17.10
РЕД ОС 7.3.7	Postgres Pro Standard 15.19.1
РЕД ОС 7.3.7	Postgres Pro Standard 16.15.1
РЕД ОС 7.3.7	Postgres Pro Standard 17.11.1
РЕД ОС 8.0.3	PostgreSQL 16.14
РЕД ОС 8.0.3	PostgreSQL 17.10
РЕД ОС 8.0.3	Postgres Pro Standard 16.15.1
РЕД ОС 8.0.3	Postgres Pro Standard 17.11.1
Astra Linux Special Edition 1.7.10.72 (Орел, Воронеж)	PostgreSQL 14.24
Astra Linux Special Edition 1.7.10.72 (Орел, Воронеж)	PostgreSQL 15.19
Astra Linux Special Edition 1.7.10.72 (Орел, Воронеж)	PostgreSQL 16.15
Astra Linux Special Edition 1.7.10.72 (Орел, Воронеж)	PostgreSQL 17.11
Astra Linux Special Edition 1.7.10.72 (Орел, Воронеж)	Postgres Pro Certified (версия ядра postgres: 14.23.1)
Astra Linux Special Edition 1.7.10.72 (Орел, Воронеж)	Postgres Pro Certified (версия ядра postgres: 15.18.1)
Astra Linux Special Edition 1.7.10.72 (Орел, Воронеж)	Postgres Pro Certified (версия ядра postgres: 16.14.1)
Astra Linux Special Edition 1.8.5.46 (Орел, Воронеж, Смоленск)	PostgreSQL 16.15
Astra Linux Special Edition 1.8.5.46 (Орел, Воронеж, Смоленск)	PostgreSQL 17.11
Astra Linux Special Edition 1.8.5.46 (Орел, Воронеж, Смоленск)	Postgres Pro Certified (версия ядра postgres: 16.14.1)
Astra Linux Special Edition 1.8.5.46 (Орел, Воронеж, Смоленск)	Postgres Pro Certified (версия ядра postgres: 17.10.1)
Ubuntu 22.04.5	PostgreSQL 14.24
Ubuntu 22.04.5	PostgreSQL 15.19
Ubuntu 22.04.5	PostgreSQL 16.15
Ubuntu 22.04.5	PostgreSQL 17.11
Ubuntu 22.04.5	Postgres Pro Standard 16.15.1
Ubuntu 22.04.5	Postgres Pro Standard 17.11.1
Ubuntu 24.04.3	PostgreSQL 16.15
Ubuntu 24.04.3	PostgreSQL 17.11

ОС	СУБД
Debian 12.15	PostgreSQL 14.24
Debian 12.15	PostgreSQL 15.19
Debian 12.15	PostgreSQL 16.15
Debian 12.15	PostgreSQL 17.11
Debian 12.15	Postgres Pro Standard 16.15.1
Debian 12.15	Postgres Pro Standard 17.11.1

1.5. Варианты поставки

Вариант поставки определяется условиями Лицензионного договора.

Состав комплекта документов на ППО, входящий во все указанные варианты поставки:

– «Руководство пользователя. Часть 1. Подсистема безопасности»;

– «Руководство пользователя. Часть 2. Подсистема «Маркет»;

– «Руководство пользователя. Часть 3. Подсистема Платформа управления»;

– «Руководство пользователя. Часть 4. Подсистема управления тенантами»;

– «Руководство пользователя. Часть 5. Подсистема Сервис уведомлений»;

– «Руководство пользователя. Часть 6. Приложение «Аврора Маркет» для операционной системы Аврора»;

– «Руководство пользователя. Часть 7. Приложение «Аврора Центр» для операционной системы Аврора»;

– «Руководство пользователя. Часть 8. Приложение «Аврора Маркет» для операционной системы Android»;

– «Руководство пользователя. Часть 9. Приложение «Аврора Центр» для операционной системы Android»;

– «Руководство пользователя. Часть 10. Приложение «Аврора Маркет» для операционных систем семейства Linux»;

– «Руководство пользователя. Часть 11. Приложение «Аврора Центр» для операционных систем семейства Linux»;

– «Руководство администратора»;

– «Рекомендации по резервному копированию»;

– «Руководство разработчика. Часть 1. Подсистема Сервис уведомлений для операционной системы Аврора»;

– «Руководство разработчика. Часть 2. Подсистема Сервис уведомлений для операционной системы Android».

1.6. Указания по эксплуатации

При эксплуатации ППО необходимо соблюдать следующие рекомендации:

– правом доступа к серверам с установленным ППО должны обладать лица, обеспечивающие функционирование информационной системы (ИС), прошедшие соответствующую подготовку, ознакомившиеся с эксплуатационной документацией (ЭД) на ППО и не рассматривающиеся в качестве нарушителей информационной безопасности;

– должны быть предусмотрены меры, исключающие возможность несанкционированного изменения аппаратной части технических средств, на которых установлено ППО;

– должна обеспечиваться периодическая (не реже 1 раза в месяц) проверка целостности программных компонентов ППО;

– на устройствах и серверах с установленным ППО должны быть реализованы меры, исключающие возможность использования средств разработки и отладчиков для редактирования кода и оперативной памяти, используемой ППО;

– должны быть обеспечены меры, исключающие возможность несанкционированной модификации программных и информационных компонентов ППО;

– физический доступ к серверам с установленной СУБД должен предоставляться лицам, включенным эксплуатирующей организацией (оператором/потребителем) ИС в перечень лиц, которые не рассматриваются в качестве нарушителя информационной безопасности;

– каналы связи, расположенные в пределах контролируемой зоны, должны быть защищены организационно-техническими мерами;

– каналы связи, расположенные за пределами контролируемой зоны, должны быть защищены с использованием средств криптографической защиты информации;

– должны быть предприняты меры по межсетевому экранированию.

При эксплуатации ППО рекомендуется применять следующие дополнительные организационные и технические меры:

– в процессе эксплуатации ППО должно быть обеспечено регулярное отслеживание наличия и осуществление установки обновлений безопасности программного обеспечения (ПО), входящего в среду функционирования ППО, в соответствии с подразделом 1.4;

– ежемесячно должен производиться поиск актуальных уязвимостей и сведений об уязвимостях ППО и среды функционирования, анализ идентифицированных уязвимостей на предмет возможности их использования для нарушения безопасности;

– в среде функционирования ППО (общесистемное ПО, указанное в подразделе 1.4) должны быть установлены все имеющиеся обновления безопасности и «патчи» для ликвидации известных уязвимостей;

– в случае обнаружения уязвимостей в ПО ППО должно производиться их устранение в соответствии с методами и процедурами, установленными предприятием-разработчиком;

– оператор с периодичностью один раз в неделю должен получать информацию о выходе обновлений ППО через службу технической поддержки предприятия-изготовителя.

2. АРХИТЕКТУРА ППО И ВАРИАНТЫ УСТАНОВКИ ППО

2.1. Описание компонентов

На рисунках (Рисунок 1, Рисунок 2) представлена физическая архитектура и диаграмма развертывания ППО.

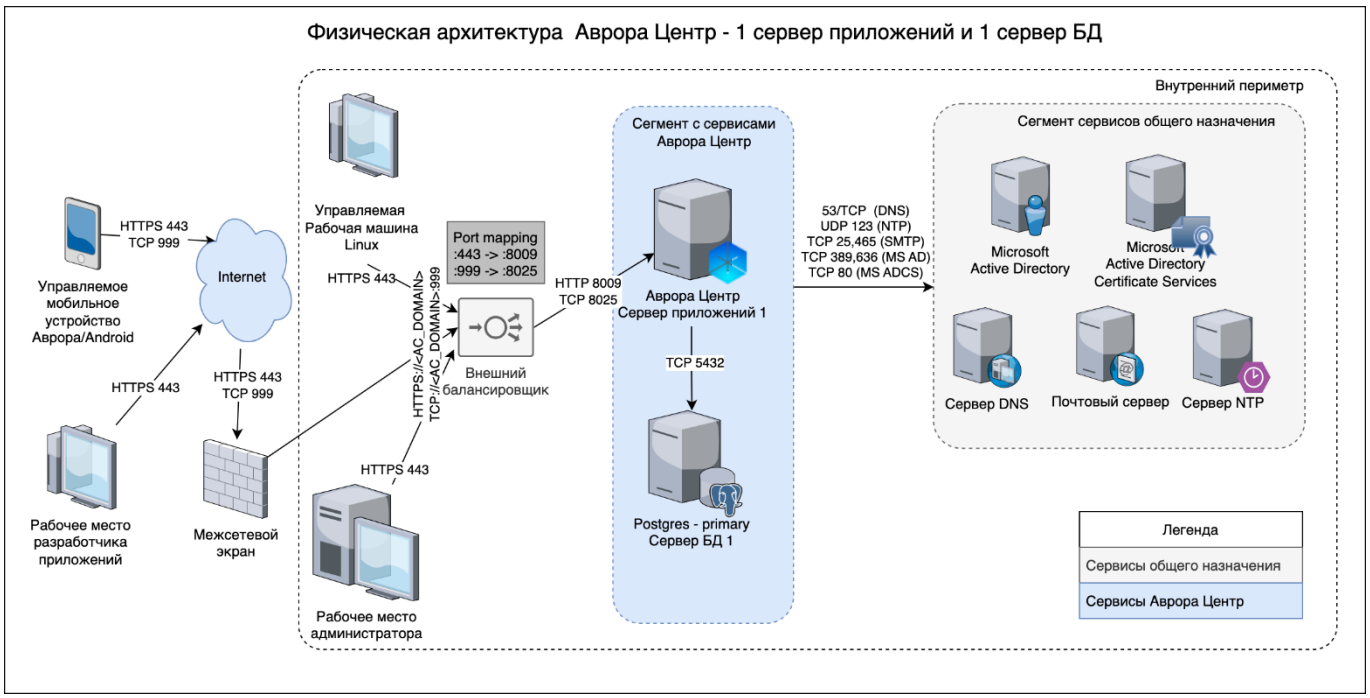


Рисунок 1

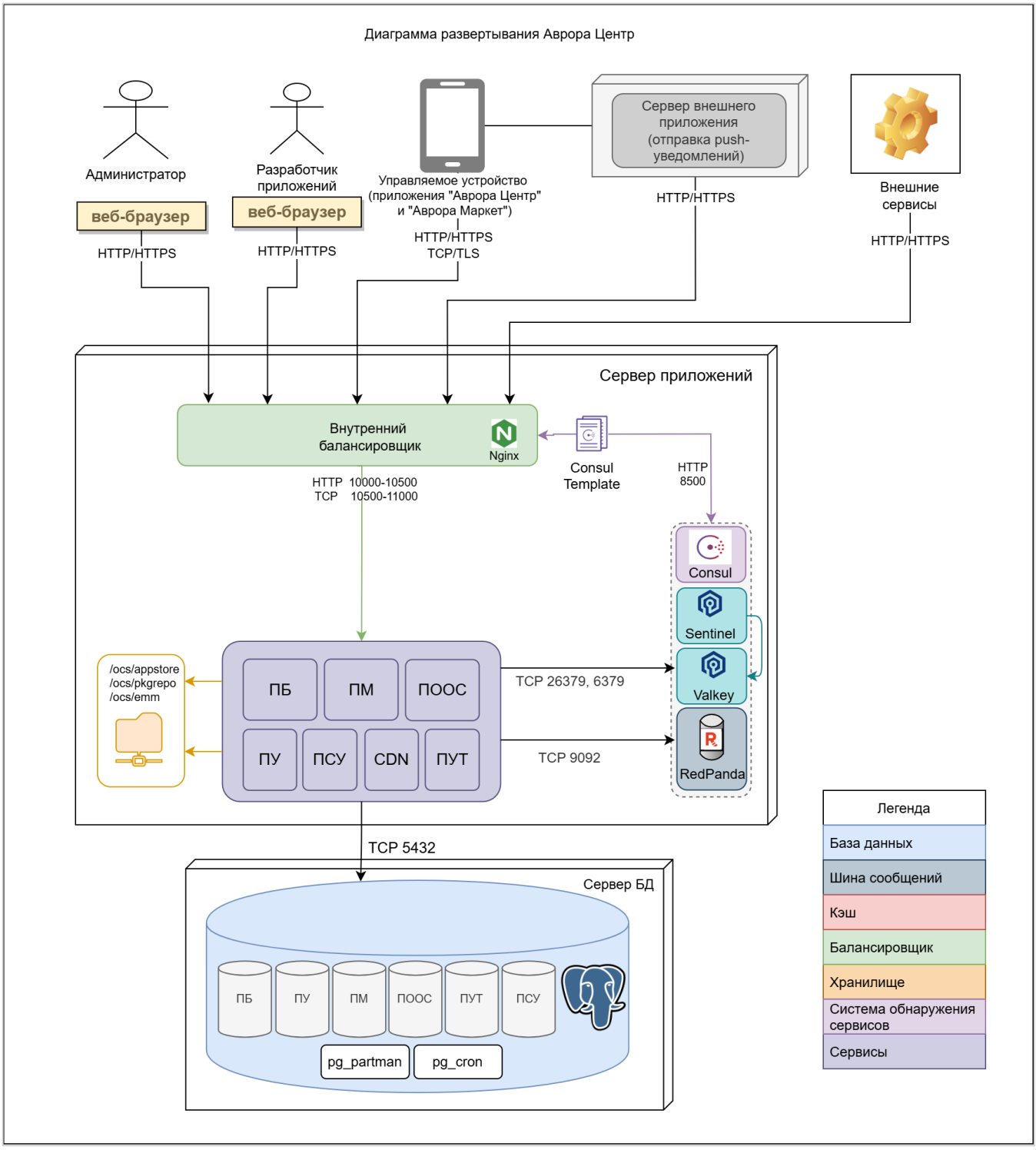

Диаграмма развертывания Аврора Центр
Администратор
веб-браузер
HTTP/HTTPS
Разработчик приложений
веб-браузер
HTTP/HTTPS
Управляемое устройство (приложения "Аврора Центр" и "Аврора Маркет")
HTTP/HTTPS
TCP/TLS
Сервер внешнего приложения (отправка push-уведомлений)
HTTP/HTTPS
Внешние сервисы
HTTP/HTTPS
Сервер приложений
Внутренний балансировщик
Nginx
HTTP 10000-10500
TCP 10500-11000
Consul Template
HTTP 8500
Consul
Sentinel
Valkey
RedPanda
/ocs/appstore
/ocs/pkgrepo
/ocs/emm
ПБ
ПМ
ПООС
ПУ
ПСУ
CDN
ПУТ
TCP 26379, 6379
TCP 9092
TCP 5432
Сервер БД
ПБ
ПУ
ПМ
ПООС
ПУТ
ПСУ
pg_partman
pg_cron
Легенда
База данных
Шина сообщений
Кэш
Балансировщик
Хранилище
Система обнаружения сервисов
Сервисы

Рисунок 2

На рисунке (Рисунок 3) представлена диаграмма развертывания ППО при интеграции с СУДИР.

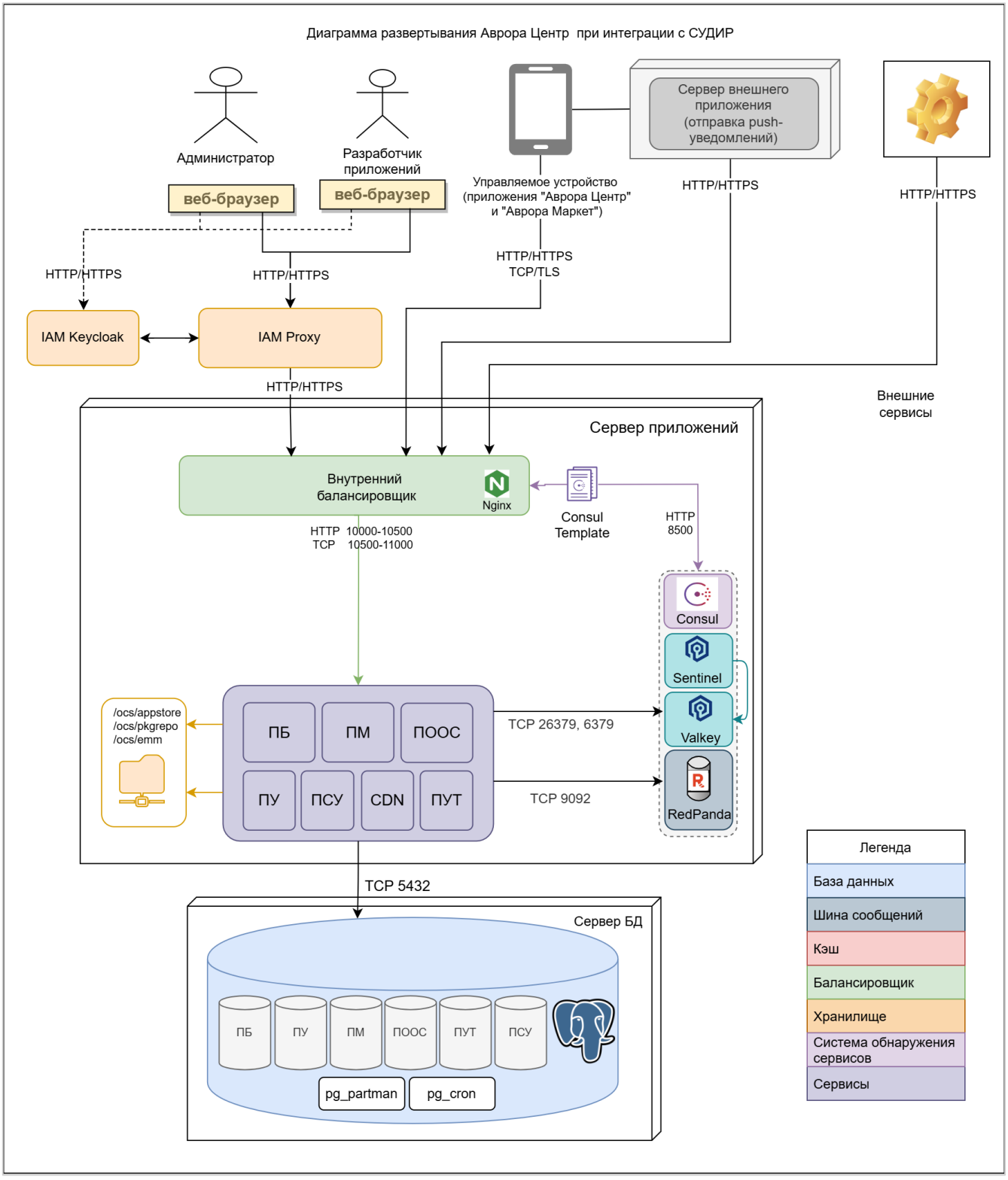


Рисунок 3

2.1.1. Сервер приложений

2.1.1.1. ППО «Аврора Центр»

ППО состоит из следующих подсистем: ПБ (`auth`), ПМ (`appstore`), ПООС (`pkgrepo`), ПУ (`emm`), ПУТ (`mt`), CDN (`cdn`), ПСУ (`push`). Каждая подсистема представляет собой набор сервисов.

Сервисы разных подсистем могут быть развернуты как на одних и тех же нодах сервера приложений, так и на отдельных. Каждый сервис работает только со своими данными, которые лежат в отдельных схемах в БД. Схемы могут быть развернуты как в общей БД, так и в разных.

Сервисы взаимодействуют между собой посредством HTTP запросов и обмена сообщениями через шину Redpanda. Все межсервисные запросы аутентифицированы.

Для корректной работы сервисов подсистем ПМ, ПУ и ПООС в кластере к нодам сервера приложений необходимо примонтировать файловое хранилище согласно п. 3.9.2.

2.1.1.2. Внутренний балансировщик

На каждом сервере приложений устанавливается внутренний балансировщик Nginx. Он выполняет функцию `reverse proxy`, служит для балансировки и передачи трафика к сервисам ППО, а также обеспечивает межсервисное взаимодействие. Для некоторых конечных точек настроено кэширование запросов к сервисам.

Nginx настраивается автоматически на основе информации о статусе сервисов из системы обнаружения сервисов Consul.

С целью защиты серверов приложений от превышения предельной нагрузки на интерфейсах, обрабатывающих внешние запросы с приложения «Аврора Центр», настроенно ограничение одновременно обрабатываемых запросов (троттлинг).

2.1.1.3. Система обнаружения сервисов Consul

Система обнаружения сервисов используется для мониторинга состояния сервисов ППО. Consul должен устанавливаться на нечетном количестве серверов.

2.1.1.4. Средство управления конфигурациями сервисов Consul Template

Средство управления конфигурациями служит для автоматической настройки распределения запросов между экземплярами сервисов за счет изменения конфигурации Nginx на основе информации о статусе сервисов ППО, получаемой из системы обнаружения сервисов Consul.

2.1.1.5. Сервис гарантированной доставки сообщений Redpanda

Сервис гарантированной доставки сообщений используется для обмена сообщениями между сервисами ППО. Сервис RedPanda должен устанавливаться на нечетном количестве серверов.

2.1.1.6. СУБД Valkey

СУБД Valkey (fork СУБД Redis) используется для хранения веб-сессий.

Для управления отказоустойчивой конфигурацией Valkey используется сервис Sentinel, который должен устанавливаться на нечетном количестве серверов.

2.1.1.7. Приложение для синхронизации файлов Syncthing

Syncthing – программное обеспечение (ПО), которое позволяет выполнять синхронизацию файлов между серверами по P2P протоколу.

ПРИМЕЧАНИЕ. Использование данного приложения является опциональным и зависит от конфигурации ППО.

2.1.1.8. Агрегатор (коллектор) для сбора метрик OpenTelemetry Collector

Агрегатор (коллектор) необходим для сбора метрик со всех нод сервера приложений ППО, их агрегации и публикации результатов для системы мониторинга Prometheus.

2.1.2. Сервер БД

В качестве сервера БД используется СУБД PostgreSQL или PostgresPro. Для работы также требуется установка расширений `pg_partman` для автоматического партиционирования и очистки накапливающихся данных, и `pg_cron` для выполнения функций в БД по расписанию.

В ненагруженных конфигурациях все данные размещаются в одном физическом инстансе БД. Внутри создаются логические БД для каждой из подсистем ППО. Данные сервисов внутри логических баз размещаются в отдельных схемах, что исключает возможность обращения одних сервисов к данным других сервисов. Описанная конфигурация снижает связность между сервисами и при необходимости позволяет вынести данные подсистем или отдельных сервисов на выделенные серверы БД.

В высоконагруженных конфигурациях рекомендуется выделить данные наиболее нагруженных подсистем ПБ (`auth`) и ПУ (`emm`) в отдельные физические базы для минимизации взаимного влияния друг на друга.

БД ПСУ (`push`) следует выносить в отдельный инстанс при высокой интенсивности запросов, более 500rps.

Сервер БД может быть установлен как с помощью сценариев установки ППО, так и самостоятельно. В сценарии установки включены минимальные возможности по установке `primary и replica` серверов, а также настройка репликации. В случае отказа основного сервера БД, переключение на резервный сервер выполняется автоматически.

2.1.3. Внешний балансировщик

Внешний балансировщик используется для распределения трафика между нодами сервера приложений. Это позволяет сбалансировать нагрузку между нодами сервера приложений и перенаправить трафик на доступные ноды в случае выхода из строя одного из серверов приложений.

Внешний балансировщик не входит в состав ППО, определяется и разворачивается пользователями самостоятельно. В директории `samples` в дистрибутиве имеется пример конфигурационного файла для балансировщика Nginx для 3-х нодовой конфигурации сервера приложений.

На внешнем балансировщике может быть настроена защита канала связи (протокол HTTPS), поддерживаются в том числе ГОСТ алгоритмы. Также существует возможность отделить консоль администратора от других интерфейсов, выделив отдельный домен (поддомен) или порт.

ВНИМАНИЕ! При установке рекомендуется в файле `nginx conf.d` добавить следующие настройки для внешнего балансировщика:

```
...
   http {
      proxy_buffers 8 16k;
      proxy_buffer_size 16k;
  }
```

Примеры настройки `nginx` для различных конфигураций, доступны в дистрибутиве: `./samples/ac-mt/nginx_external-balancer/conf.d`.

Пример файла настройки `nginx` для контент-сервера: `./samples/ac-mt/nginx_external-balancer/conf.d/content-server.conf`.

2.1.4. Внешние службы

В данном пункте приведены описания внешних служб, которые не устанавливаются вместе с основными компонентами.

2.1.4.1. Сервер DNS

Сервер DNS используется для получения информации о доменах (IP-адреса по имени хоста ЭВМ или устройства).

2.1.4.2. Сервер NTP

Сервер NTP используется для автоматической синхронизации времени на устройстве при добавлении правила политики «Конфигурация/Синхронизация времени». Адреса NTP серверов необходимо указывать при развертывании системы в конфигурационном файле для ПУ.

Пример:

```
timeServerSettings:
    hosts:
      - ntp1.vniiftri.ru
      - ntp2.niiftri.irkutsk.ru
      - vniiftri2.khv.ru
```

ПРИМЕЧАНИЕ. В примере указаны стандартные адреса NTP-серверов, которые находятся в конфигурационном файле. При необходимости их можно заменить на другие.

2.1.4.3. Почтовый сервер

Почтовый сервер используется для рассылки кодов активации устройств.

2.1.4.4. Службы каталогов

Интеграция ППО со службами каталогов (Microsoft Active Directory, ALD Pro, Samba DC) используется для автоматической синхронизации списка пользователей устройств в ПУ со списком пользователей организации, а также для автоматической привязки устройства к пользователю.

2.1.4.5. Службы сертификатов Active Directory (Active Directory Certificate Services)

Службы сертификатов Active Directory используются для управления ключевой информацией (закрытый ключ, открытый ключ, сертификат открытого ключа) пользователей устройств.

2.1.5. Сервер внешнего приложения

Сервер внешнего приложения является серверной частью произвольного внешнего (по отношению к ППО) приложения, которое с помощью ПСУ осуществляет передачу push-уведомлений на соответствующее мобильное приложение.

2.1.6. Перечень используемых внешних репозиториев

В таблице (Таблица 6) приведен список внешних репозиториев, которые используются при установке ППО.

Таблица 6

Назначение ресурса	Ссылка на ресурс
Официальный репозиторий Python Packaging Authority (PyPA) для начальной загрузки и установки инструментов управления пакетами, таких как `pip`, `setuptools` и `wheel`	https://bootstrap.pypa.io/
Хранилище ПО для языка программирования Python	https://pypi.org/
Хранилище коллекций готовых модулей и плагинов для Ansible	https://github.com/ansible-collections

ПРИМЕЧАНИЕ. Если данные ресурсы недоступны по причине закрытого контура, тогда необходимо обратиться в службу технической поддержки предприятия-изготовителя ППО и запросить архив с зависимостями для установки ППО без доступа к указанным ресурсам.

2.2. Внешние интерфейсы сервера приложений ППО

Перед нодами сервера приложений могут располагаться различные компоненты сетевой инфраструктуры (например, межсетевой экран, внешний балансировщик, средство криптографической защиты информации и др.), которые обрабатывают поступающие к серверу приложений запросы.

По умолчанию ноды сервера приложений для обработки внешних запросов используют специальные выделенные порты, взаимодействие с которыми осуществляется по не защищенному протоколу.

Для того чтобы ППО было доступно из внешней сети, на компонентах сетевой инфраструктуры необходимо настроить переадресацию портов.

Пример таблицы переадресации портов (port mapping) приведен на физической архитектуре, а также в таблице (Таблица 7).

Таблица 7

Назначение порта	Порт сервера приложений	Порт СКЗИ
Обработка запросов консолей пользователей/администраторов, а также запросов приложений «Аврора Центр» и «Маркет»	8009 (http)	443 (HTTPS)
Обработка запросов контент-серверов	8024 (http)	8443 (HTTPS)
Обработка запросов от устройств к ПСУ для получения push-уведомлений	8025 (tcp)	999 (tls)

Для обращения к ППО имеет смысл завести отдельный домен `<AC_DOMAIN>` и назначить его на самый первый компонент сетевой инфраструктуры, который принимает запросы от пользователей ППО и устройств.

Домен `<AC_DOMAIN>` и внешние порты должны быть указаны в соответствующих конфигурационных файлах ППО в процессе его настройки.

2.3. Варианты установки ППО

В зависимости от требований к количеству поддерживаемых устройств применяются различные варианты установки ППО.

2.3.1. Требования к аппаратным характеристикам

Для запуска ППО и СУБД на 1 сервере требуются следующие минимальные аппаратные характеристики:

- 4 ядра процессора;
- 8 ГБ оперативной памяти;
- 50 ГБ свободного места на жестком диске.

Данную конфигурацию рекомендуется использовать в качестве тестового стенда для ознакомления с функционалом ППО и иных случаях, где не предъявляются требования к производительности.

ВНИМАНИЕ! Процессоры серверов с сервисом гарантированной доставки сообщений RedPanda должны поддерживать набор инструкций SSE 4.2 (Streaming SIMD Extensions 4.2).

ПРИМЕЧАНИЯ:

✓ В таблицах ниже указаны аппаратные требования к ЭВМ в зависимости от максимального количества поддерживаемых устройств.

✓ При размещении файлового хранилища на сервере приложений, объем жесткого диска сервера приложений необходимо увеличить на размер данного хранилища.

В таблице (Таблица 8) приведены аппаратные характеристики ЭВМ, для работы с 10 000 устройств.

Таблица 8

Сервера	Параметры для 10 000 устройств					
	Процессор, количество ядер	Объем оперативной памяти, ГБ	Объем жесткого диска SSD, ГБ	iops	Скорость сети, Гбит/с	Количество серверов
Сервер приложений	4	10	75	200	0,1	3
Сервер БД ПБ, ПМ, ПУ, ПУТ, ПООС, ПСУ	4	4	700	200	0,1	2
Внешний балансировщик (nginx)	2	2	20	30	0,1	2

В таблице (Таблица 9) приведены аппаратные характеристики ЭВМ, для работы с 50 000 устройств.

Таблица 9

Сервера	Параметры для 50 000 устройств					
	Процессор, количество ядер	Объем оперативной памяти, ГБ	Объем жесткого диска SSD, ГБ	iops	Скорость сети, Гбит/с	Количество серверов
Сервер приложений	6	12	100	600	0,1	3
Сервер БД ПБ, ПМ, ПУ, ПУТ, ПООС, ПСУ	6	6	3200	200	0,1	2
Content Server	2	4	150	100	1	2
Внешний балансировщик (nginx)	4	4	30	30	0,1	2
Внешний балансировщик content (nginx)	2	2	20	30	1	2

В таблице (Таблица 10) приведены аппаратные характеристики ЭВМ, для работы со 100 000 устройств.

Таблица 10

Сервера	Параметры для 100 000 устройств					
	Процессор, количество ядер	Объем оперативной памяти, ГБ	Объем жесткого диска SSD, ГБ	iops	Скорость сети, Гбит/с	Количество серверов
Сервер приложений	10	14	110	100	0,2	3
Сервер БД ПБ, ПМ, ПУ, ПУТ, ПООС, ПСУ	8	8	6300	600	0,2	2
Сервер инфраструктурных компонентов	2	8	50	1000	0,1	3
Content Server	2	4	150	100	2,5	4
Внешний балансировщик (nginx)	4	6	30	30	0,2	4
Внешний балансировщик content (nginx)	2	4	20	30	2,5	4

В таблице (Таблица 11) приведены аппаратные характеристики ЭВМ, на для работы с 200 000 устройств.

Таблица 11

Сервера	Параметры для 200 000 устройств					
	Процессор, количество ядер	Объем оперативной памяти, ГБ	Объем жесткого диска SSD, ГБ	iops	Скорость сети, Гбит/с	Количество серверов
Сервер приложений	16	16	130	100	0,2	3
Сервер БД ПБ	4	12	7000	400	0,2	2
Сервер БД ПМ, ПУ, ПУТ, ПООС, ПСУ	12	12	7000	800	0,2	2
Сервер инфраструктурных компонентов	4	10	70	2000	0,1	3
Content Server	2	6	200	200	5	6
Внешний балансировщик (nginx)	4	8	30	30	0,2	6
Внешний балансировщик content (nginx)	4	4	20	30	5	6

В таблице (Таблица 12) приведены аппаратные характеристики ЭВМ, для работы с 500 000 устройств.

Таблица 12

Сервера	Параметры 500 000 устройств					
	Процессор, количество ядер	Объем оперативной памяти, ГБ	Объем жесткого диска SSD, ГБ	iops	Скорость сети, Гбит/с	Количество серверов
Сервер приложений	12	18	160	100	0,2	6
Сервер БД ПБ	6	22	16000	1000	0,2	2
Сервер БД ПМ, ПУ, ПУТ, ПООС, ПСУ	20	24	15000	2000	0,2	2

Сервера	Параметры 500 000 устройств					
	Процессор, количество ядер	Объем оперативной памяти, ГБ	Объем жесткого диска SSD, ГБ	iops	Скорость сети, Гбит/с	Количество серверов
Сервер БД ПСУ	20	24	2000	200	0,2	2
Сервер инфраструктурных компонентов	4	12	70	3000	0,1	3
Content Server	4	8	200	200	5	6
Внешний балансировщик (nginx)	6	10	50	50	0,2	6
Внешний балансировщик content (nginx)	4	4	30	50	5	6

Далее приведены схемы установки ППО.

2.3.2. Конфигурация все в одном (ППО и СУБД на одном сервере)

Сервисы ППО, инфраструктурные компоненты ППО, компоненты среды функционирования ППО, а также сервер БД установлены на одном сервере (Рисунок 4).

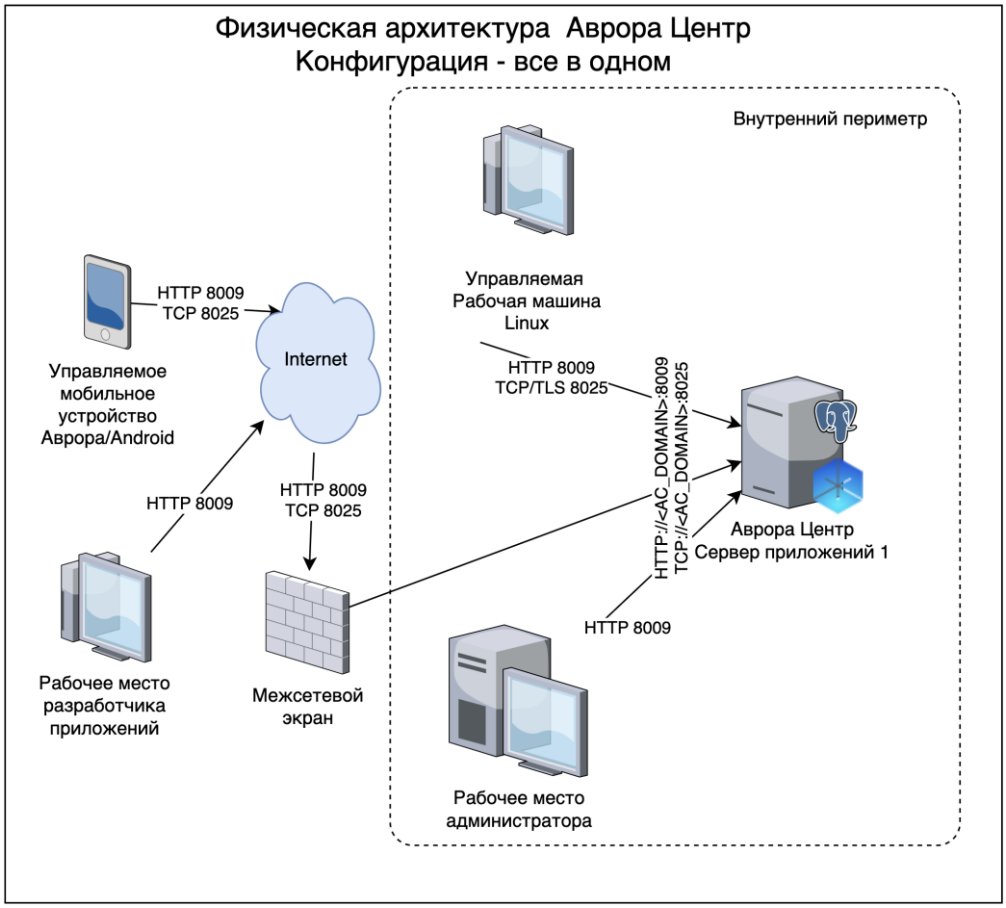


Рисунок 4

ПРИМЕЧАНИЕ. Внешний балансировщик и СКЗИ не используются.

Данную конфигурацию рекомендуется использовать для ознакомления с функционалом ППО и иных случаях, где не предъявляются требования к производительности, безопасности и надежности (Рисунок 5).

ПРИМЕЧАНИЕ. Данный вариант развертывания (включая запуск из единого docker-образа) не имеет поддержки отказоустойчивости. При аппаратном или программном сбое единственного узла произойдет полная остановка работы всех сервисов ППО.

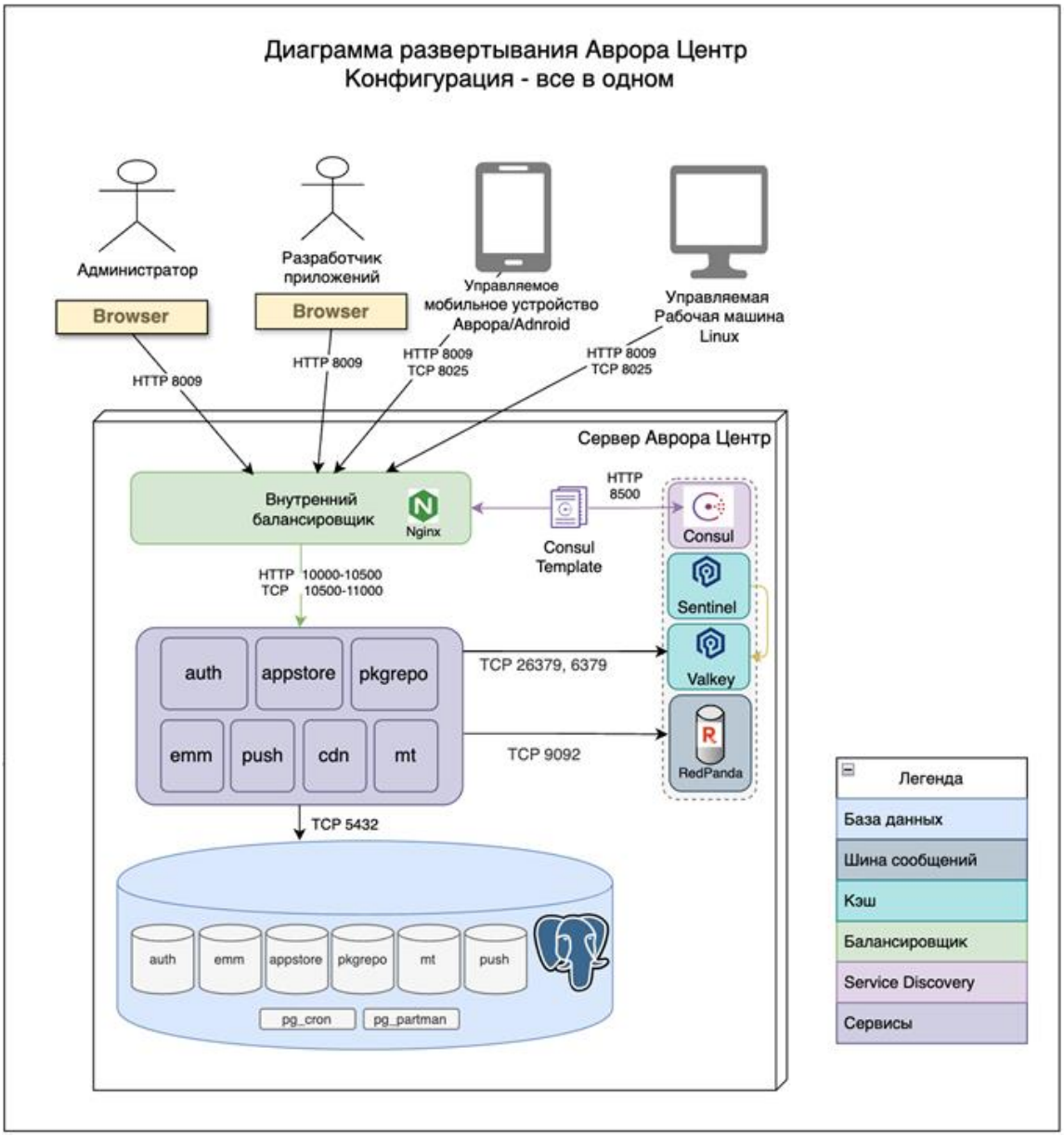


Рисунок 5

2.3.3. Конфигурация из одного сервера приложений и одного сервера БД

Сервисы ППО и инфраструктурные компоненты ППО установлены на одном сервере, а сервер БД установлен на отдельном сервере. Также настроены внешний балансировщик, на котором задан внешний домен ППО, и СКЗИ для защиты канала связи.

Данную конфигурацию рекомендуется использовать в качестве тестовой, либо в случаях, когда не предъявляются требования к отказоустойчивости и производительности (Рисунок 6, Рисунок 7).

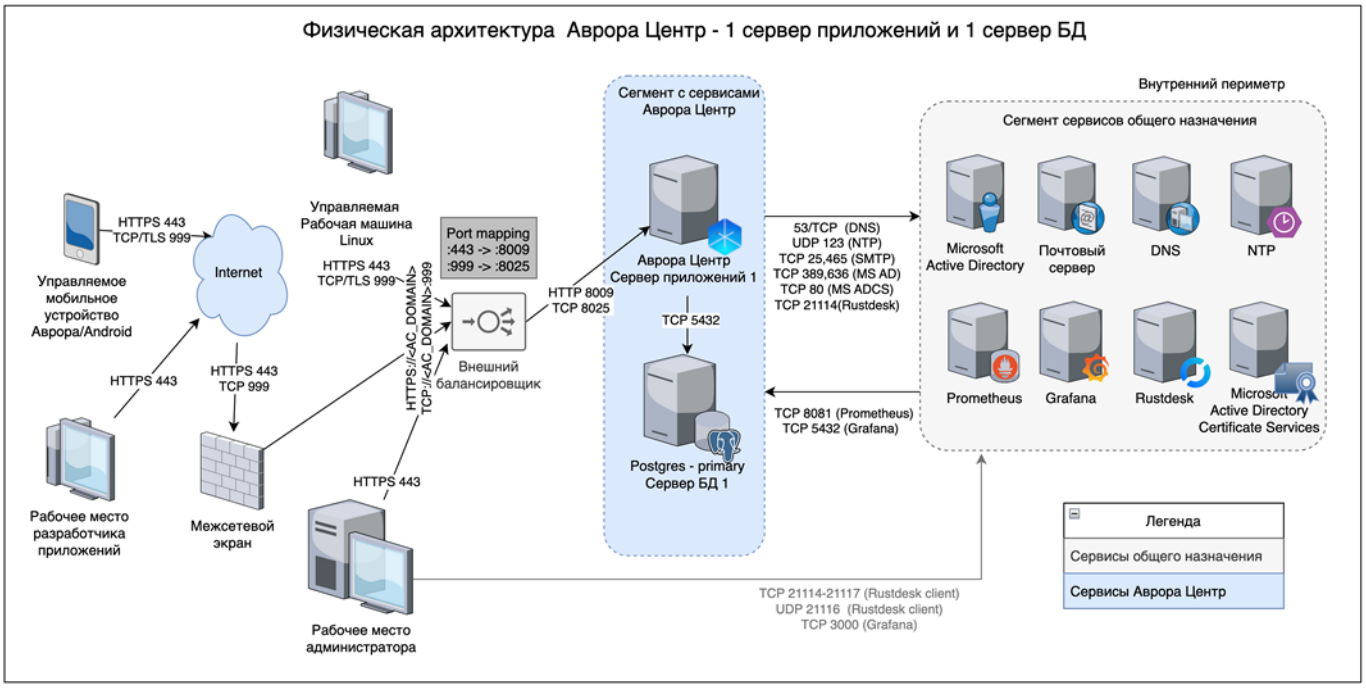

Физическая архитектура Аврора Центр - 1 сервер приложений и 1 сервер БД
Управляемое мобильное устройство Аврора/Android
HTTPS 443
TCP/TLS 999
Internet
Управляемая Рабочая машина Linux
HTTPS 443
TCP/TLS 999
Port mapping
:443 -> :8009
:999 -> :8025
HTTPS://<AC_DOMAIN>
TCP://<AC_DOMAIN>:999
Внешний балансировщик
HTTP 8009
TCP 8025
Сегмент с сервисами Аврора Центр
Аврора Центр Сервер приложений 1
TCP 5432
Postgres - primary Сервер БД 1
53/TCP (DNS)
UDP 123 (NTP)
TCP 25,465 (SMTP)
TCP 389,636 (MS AD)
TCP 80 (MS ADCS)
TCP 21114(Rustdesk)
TCP 8081 (Prometheus)
TCP 5432 (Grafana)
Внутренний периметр
Сегмент сервисов общего назначения
Microsoft Active Directory
Почтовый сервер
DNS
NTP
Prometheus
Grafana
Rustdesk
Microsoft Active Directory Certificate Services
HTTPS 443
TCP 999
HTTPS 443
Рабочее место разработчика приложений
Межсетевой экран
HTTPS 443
Рабочее место администратора
TCP 21114-21117 (Rustdesk client)
UDP 21116 (Rustdesk client)
TCP 3000 (Grafana)
Легенда
Сервисы общего назначения
Сервисы Аврора Центр

Рисунок 6

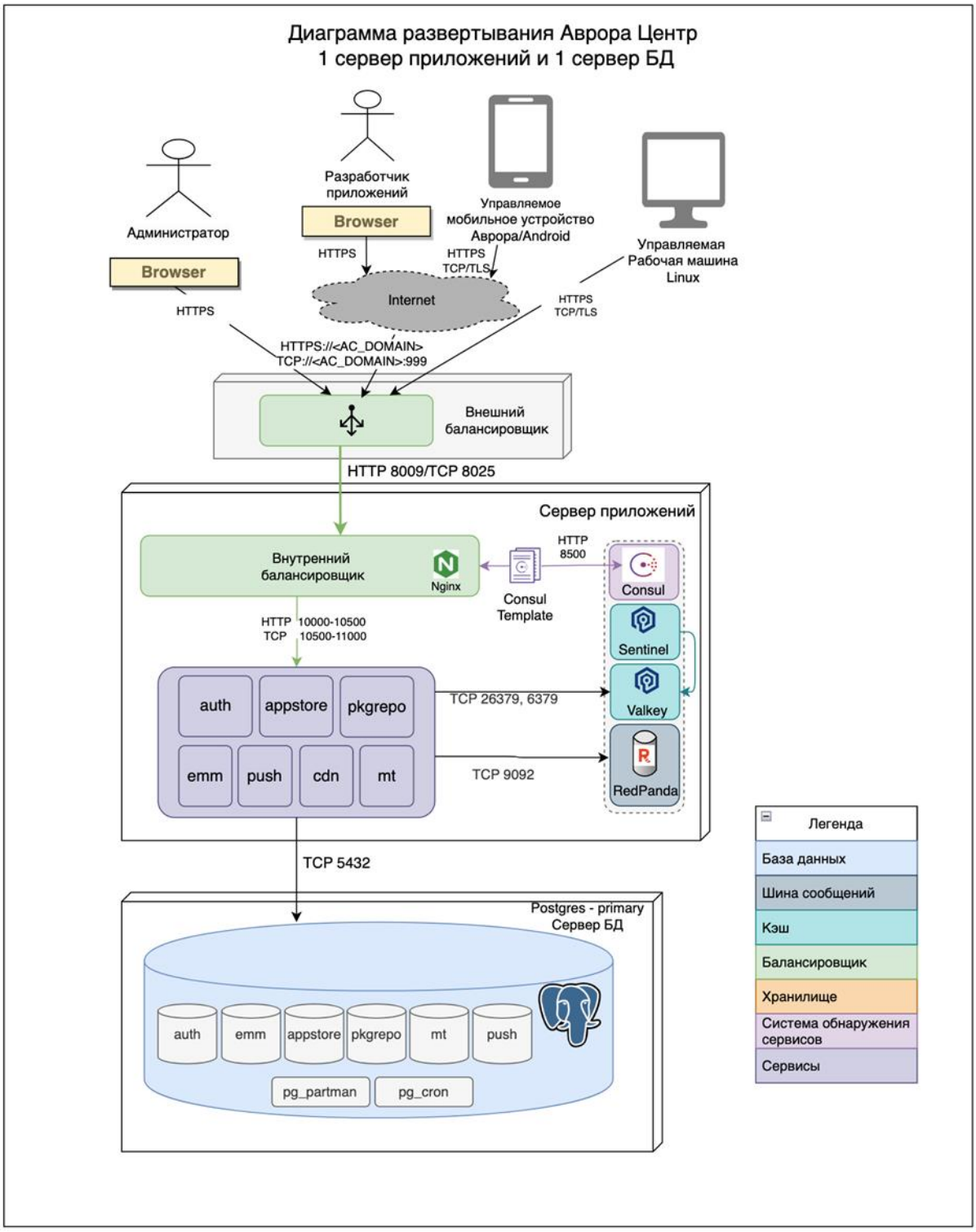

Диаграмма развертывания Аврора Центр
1 сервер приложений и 1 сервер БД
Разработчик приложений
Browser
Управляемое мобильное устройство Аврора/Android
Управляемая Рабочая машина Linux
Администратор
Browser
HTTPS
HTTPS TCP/TLS
HTTPS
HTTPS TCP/TLS
HTTPS
Internet
HTTPS://<AC_DOMAIN>
TCP://<AC_DOMAIN>:999
Внешний балансировщик
HTTP 8009/TCP 8025
Сервер приложений
Внутренний балансировщик
Nginx
HTTP 8500
Consul
Consul Template
Sentinel
HTTP 10000-10500
TCP 10500-11000
auth
appstore
pkgrepo
emm
push
cdn
mt
TCP 26379, 6379
Valkey
TCP 9092
RedPanda
TCP 5432
Postgres - primary
Сервер БД
auth
emm
appstore
pkgrepo
mt
push
pg_partman
pg_cron
Легенда
База данных
Шина сообщений
Кэш
Балансировщик
Хранилище
Система обнаружения сервисов
Сервисы

Рисунок 7

2.3.4. Кластерная конфигурация (поддержка до 10000 устройств)

В отличии от предыдущей конфигурации в данной конфигурации для обеспечения отказоустойчивости сервисы ППО и инфраструктурные компоненты ППО собираются в кластер из трех нод, каждая из которых обрабатывает запросы. Также устанавливается резервный сервер БД.

Данную схему установки рекомендуется использовать в качестве отказоустойчивой конфигурации, поддерживающей до 10000 устройств (Рисунок 8, Рисунок 9).

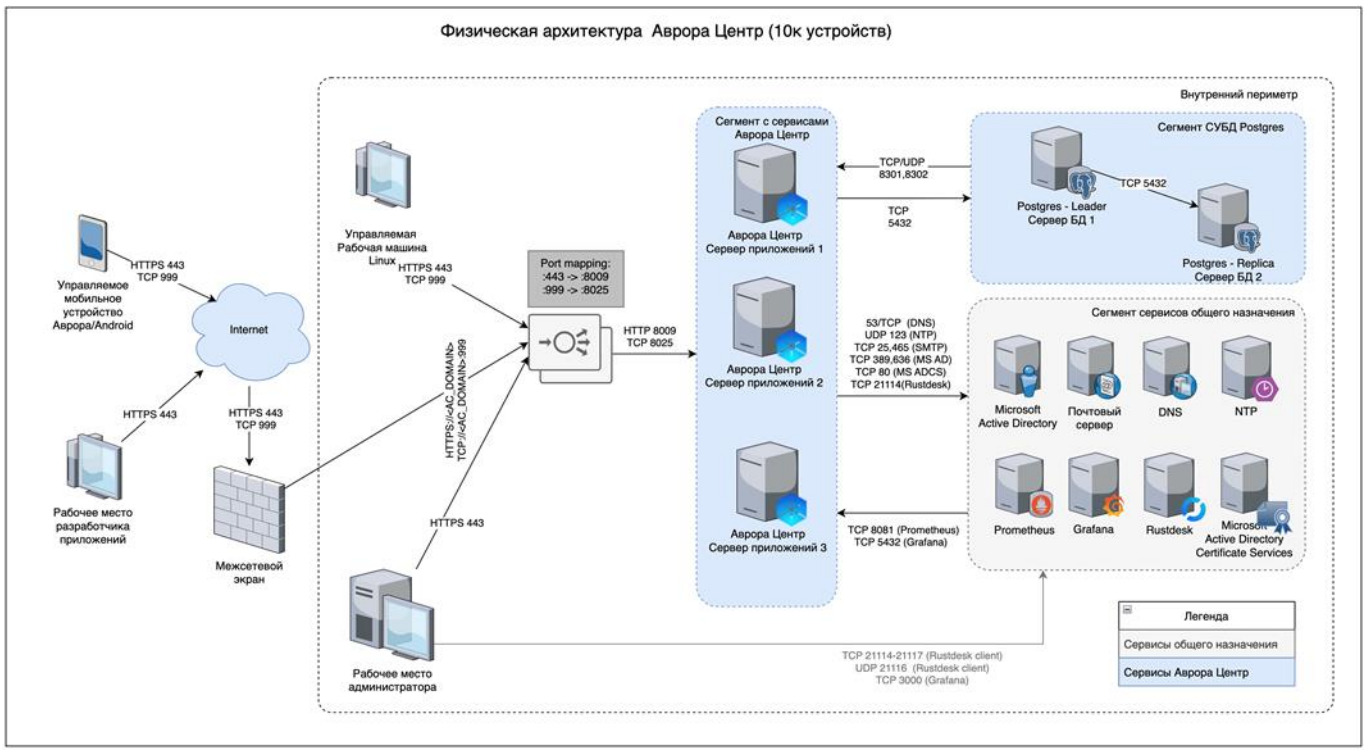


Рисунок 8

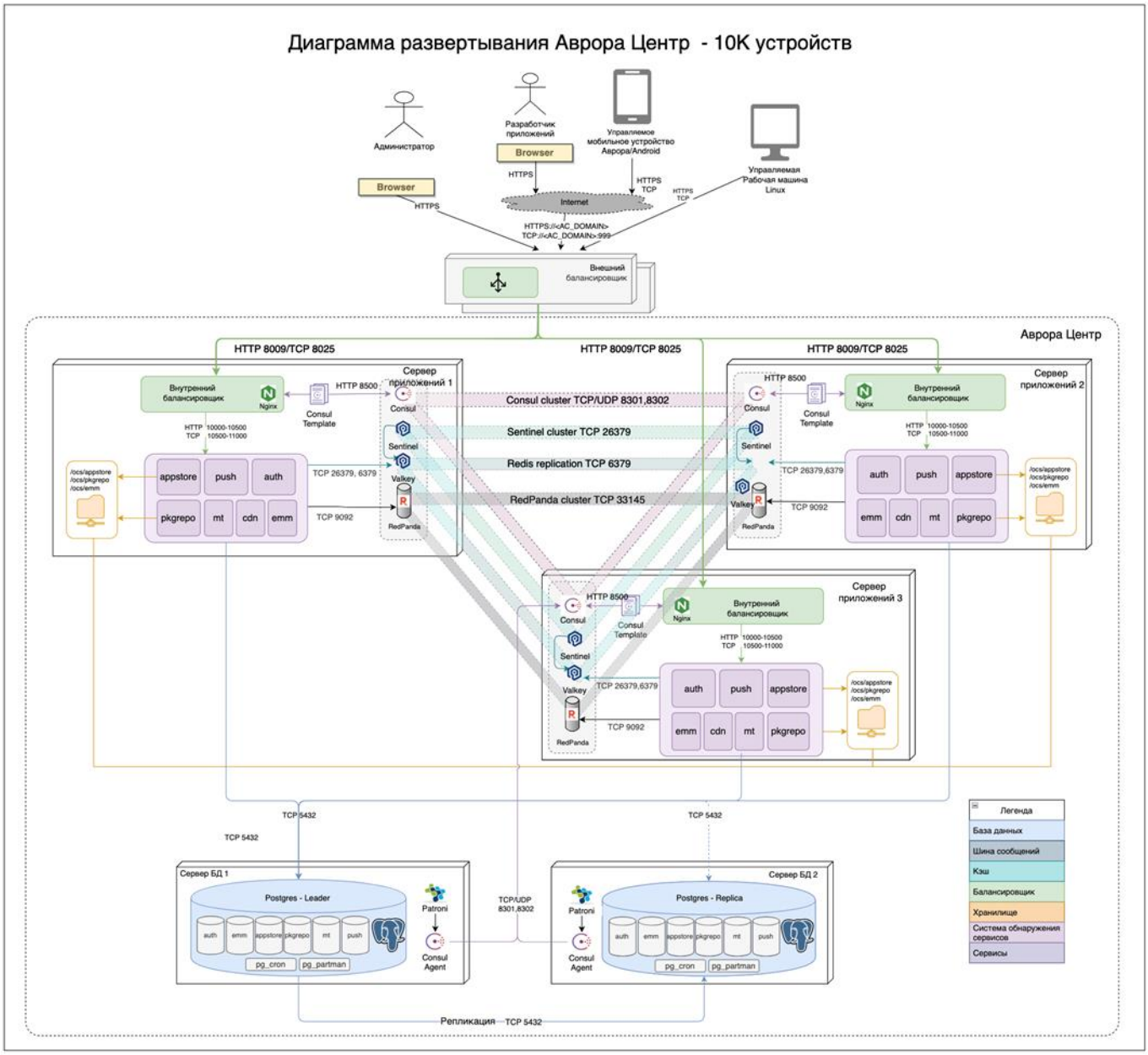


Рисунок 9

2.3.5. Кластерная конфигурация с контент-серверами (поддержка до 100000 устройств)

В отличие от конфигурации, поддерживающей до 10000 устройств, в данной конфигурации для оптимизации доставки контента и снижения нагрузки на сервер приложений используются контент-серверы.

Данную конфигурацию рекомендуется использовать, когда требуется поддержка до 100000 устройств и/или в случае большой территориальной удаленности устройств от сервера приложений (Рисунок 10, Рисунок 11).

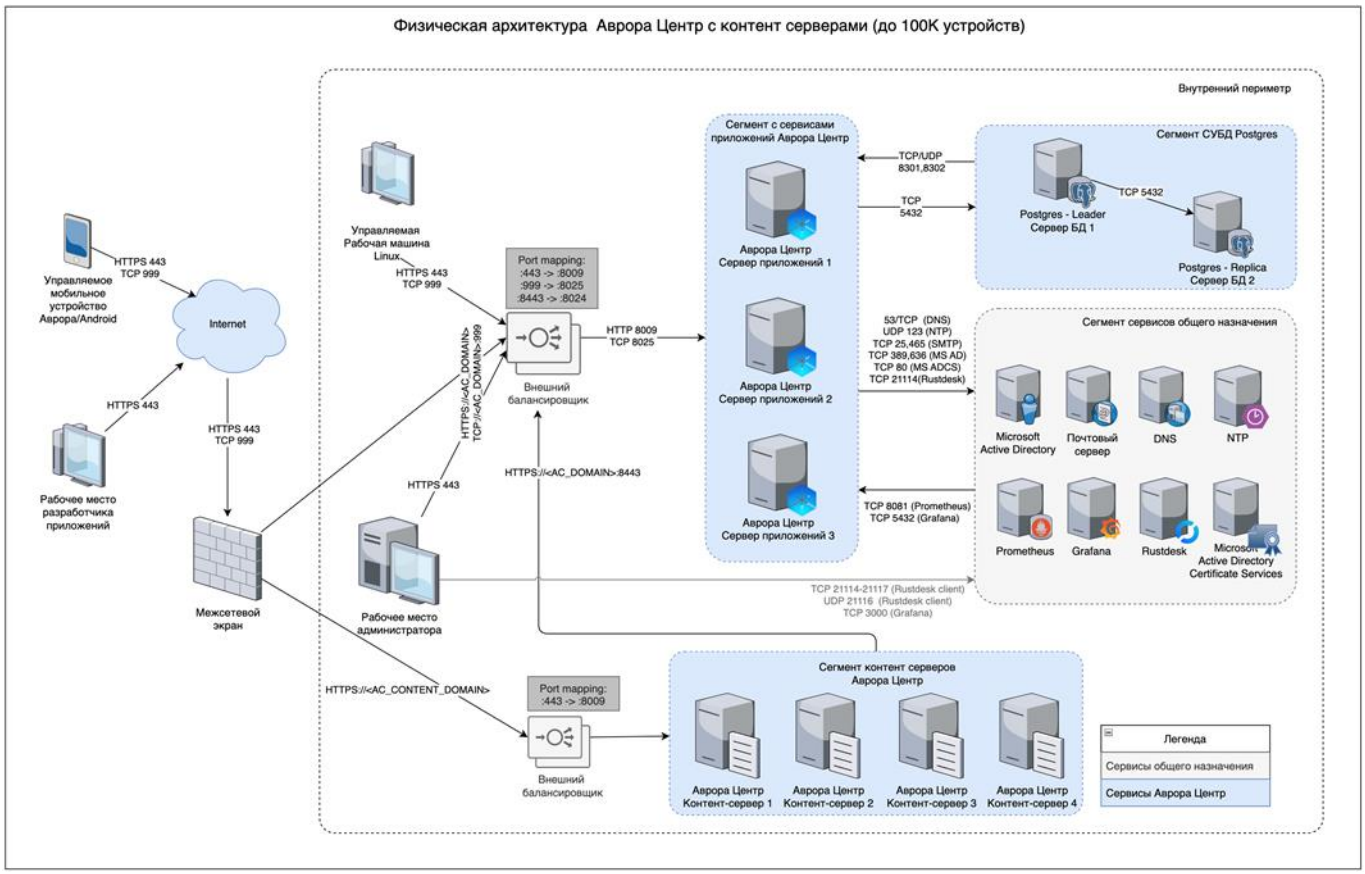


Рисунок 10

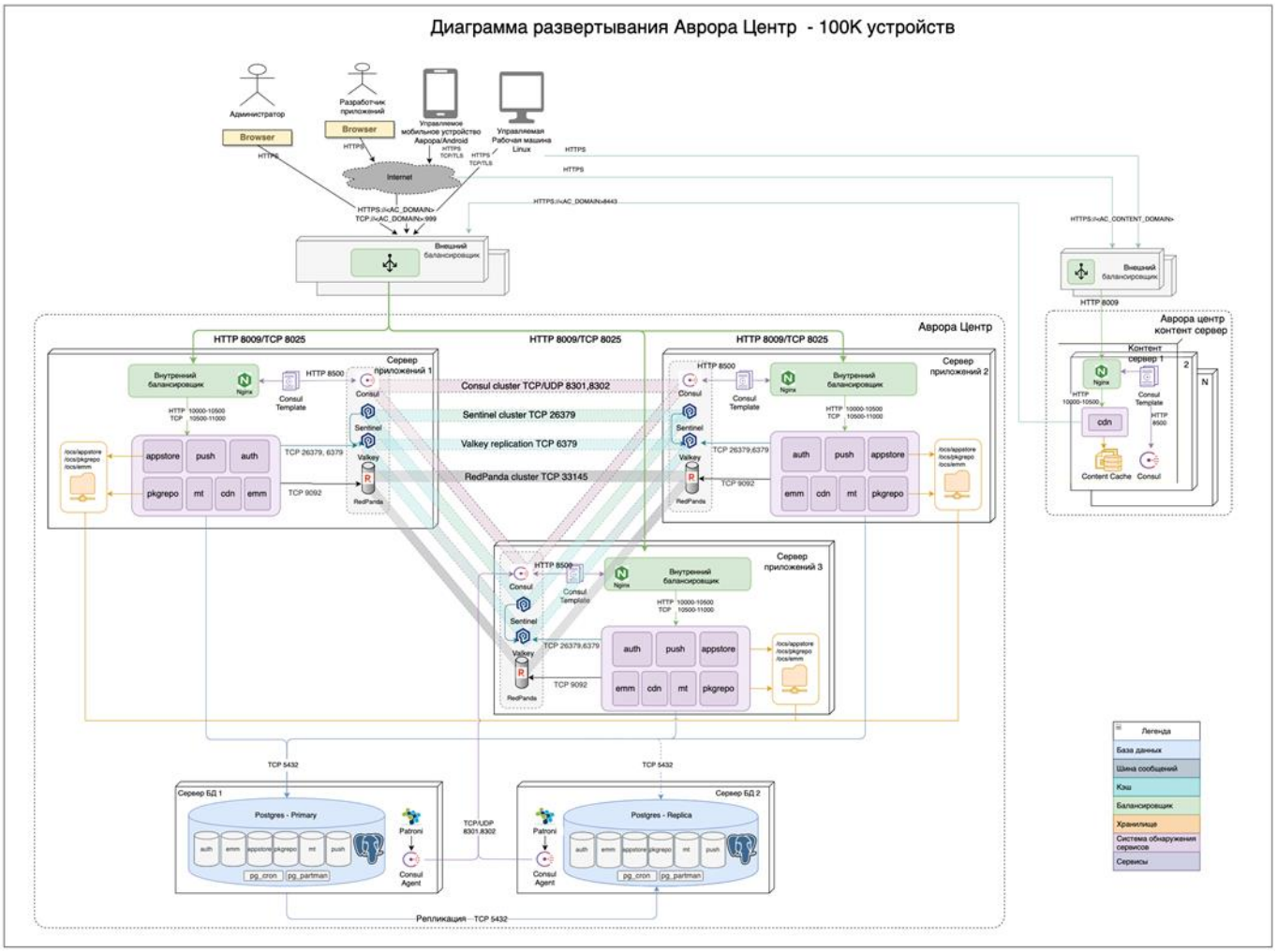


Рисунок 11

2.3.6. Кластерная конфигурация с контент-сервером и раздельными серверами БД (поддержка до 500000 устройств)

В данной конфигурации для минимизации взаимного влияния друг на друга осуществляется разделение наиболее нагруженных БД ПБ (`auth`) и ПУ (`emm`) по отдельным серверам. Инфраструктурные компоненты ППО устанавливаются на отдельных серверах, сервисы ППО собираются в кластер из шести нод. Контент-серверы можно разместить на отдельных площадках (например, в разных регионах).

Данную конфигурацию рекомендуется использовать для поддержки максимального количества устройств (до 500000 устройств) (Рисунок 12, Рисунок 13).

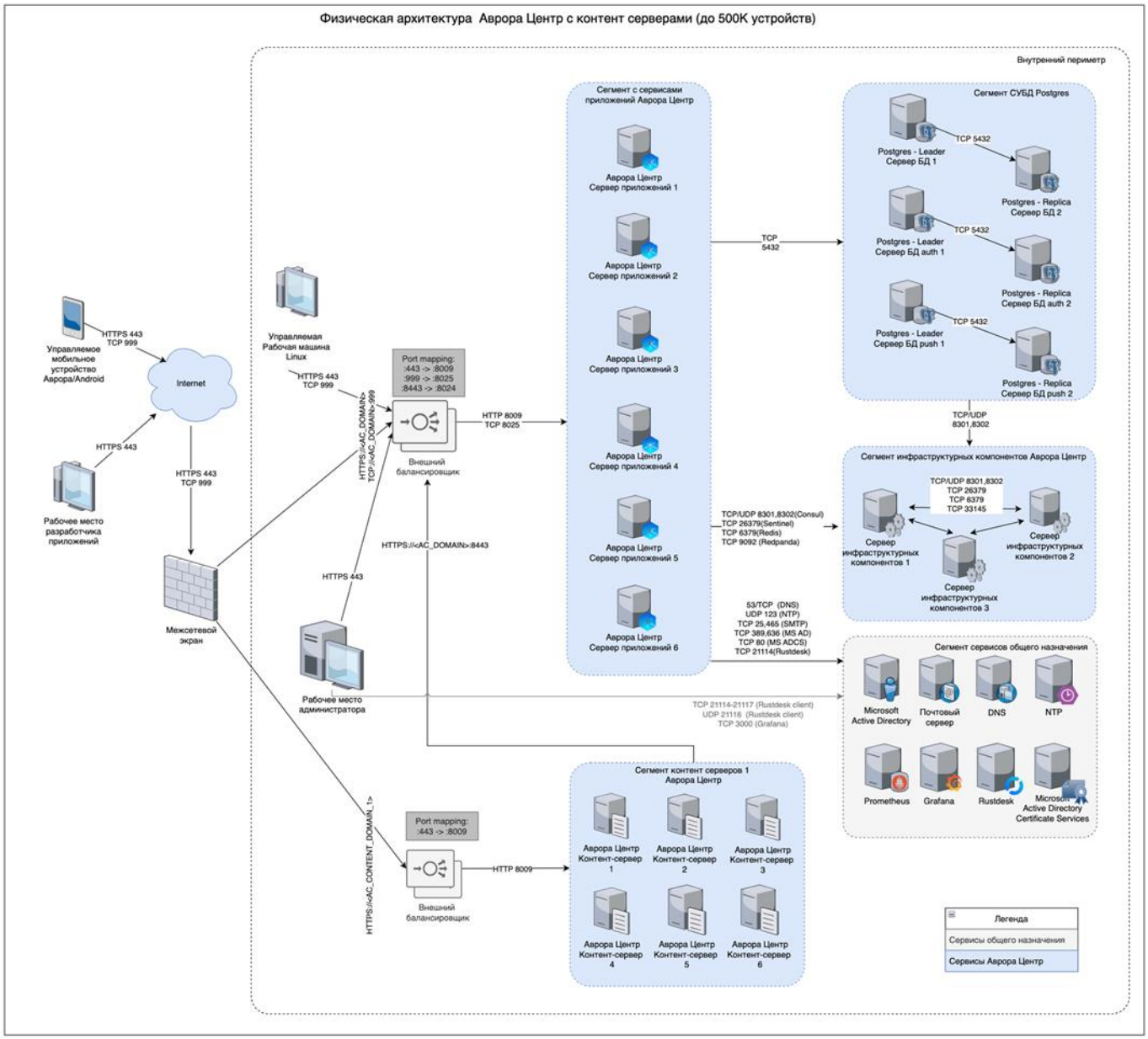

Физическая архитектура Аврора Центр с контент серверами (до 500К устройств)
Внутренний периметр
Управляемое мобильное устройство Аврора/Android
HTTPS 443
TCP 999
Internet
HTTPS 443
Рабочее место разработчика приложений
HTTPS 443
TCP 999
Межсетевой экран
Управляемая Рабочая машина Linux
HTTPS 443
TCP 999
HTTPS://<AC_DOMAIN>
TCP://<AC_DOMAIN>:999
Port mapping:
:443 -> :8009
:999 -> :8025
:8443 -> :8024
Внешний балансировщик
HTTP 8009
TCP 8025
HTTPS://<AC_DOMAIN>:8443
HTTPS 443
Рабочее место администратора
Сегмент с сервисами приложений Аврора Центр
Аврора Центр Сервер приложений 1
Аврора Центр Сервер приложений 2
Аврора Центр Сервер приложений 3
Аврора Центр Сервер приложений 4
Аврора Центр Сервер приложений 5
Аврора Центр Сервер приложений 6
TCP 5432
Сегмент СУБД Postgres
Postgres - Leader Сервер БД 1
TCP 5432
Postgres - Replica Сервер БД 2
Postgres - Leader Сервер БД auth 1
TCP 5432
Postgres - Replica Сервер БД auth 2
Postgres - Leader Сервер БД push 1
TCP 5432
Postgres - Replica Сервер БД push 2
TCP/UDP 8301,8302
Сегмент инфраструктурных компонентов Аврора Центр
TCP/UDP 8301,8302
TCP 26379
TCP 6379
TCP 33145
Сервер инфраструктурных компонентов 1
Сервер инфраструктурных компонентов 2
Сервер инфраструктурных компонентов 3
TCP/UDP 8301,8302(Consul)
TCP 26379(Sentinel)
TCP 6379(Redis)
TCP 9092 (Redpanda)
53/TCP (DNS)
UDP 123 (NTP)
TCP 25,465 (SMTP)
TCP 389,636 (MS AD)
TCP 80 (MS ADCS)
TCP 21114(Rustdesk)
Сегмент сервисов общего назначения
Microsoft Active Directory
Почтовый сервер
DNS
NTP
Prometheus
Grafana
Rustdesk
Microsoft Active Directory Certificate Services
TCP 21114-21117 (Rustdesk client)
UDP 21116 (Rustdesk client)
TCP 3000 (Grafana)
Сегмент контент серверов 1 Аврора Центр
Аврора Центр Контент-сервер 1
Аврора Центр Контент-сервер 2
Аврора Центр Контент-сервер 3
Аврора Центр Контент-сервер 4
Аврора Центр Контент-сервер 5
Аврора Центр Контент-сервер 6
HTTPS://<AC_CONTENT_DOMAIN_1>
Port mapping:
:443 -> :8009
Внешний балансировщик
HTTP 8009
Легенда
Сервисы общего назначения
Сервисы Аврора Центр

Рисунок 12

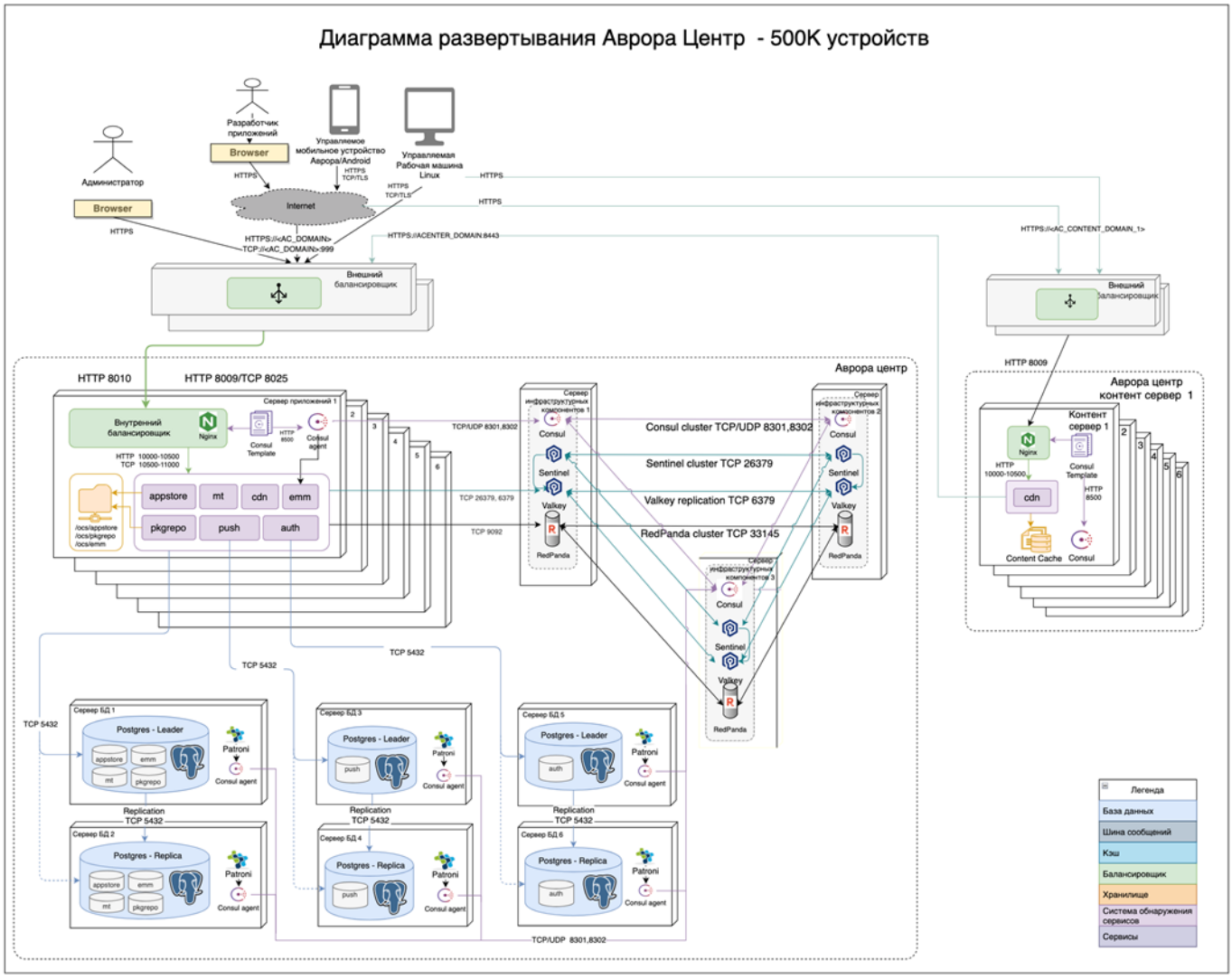


Рисунок 13

2.3.7. Катастрофоустойчивая кластерная конфигурация с установкой серверов приложений и серверов БД в двух центрах обработки данных

В данной конфигурации с целью защиты от природных, техногенных катастроф или терактов и обеспечения непрерывности бизнес-процессов установка серверов приложений и серверов БД осуществляется в двух центрах обработки данных (ЦОД) – основной и резервный (Рисунок 14).

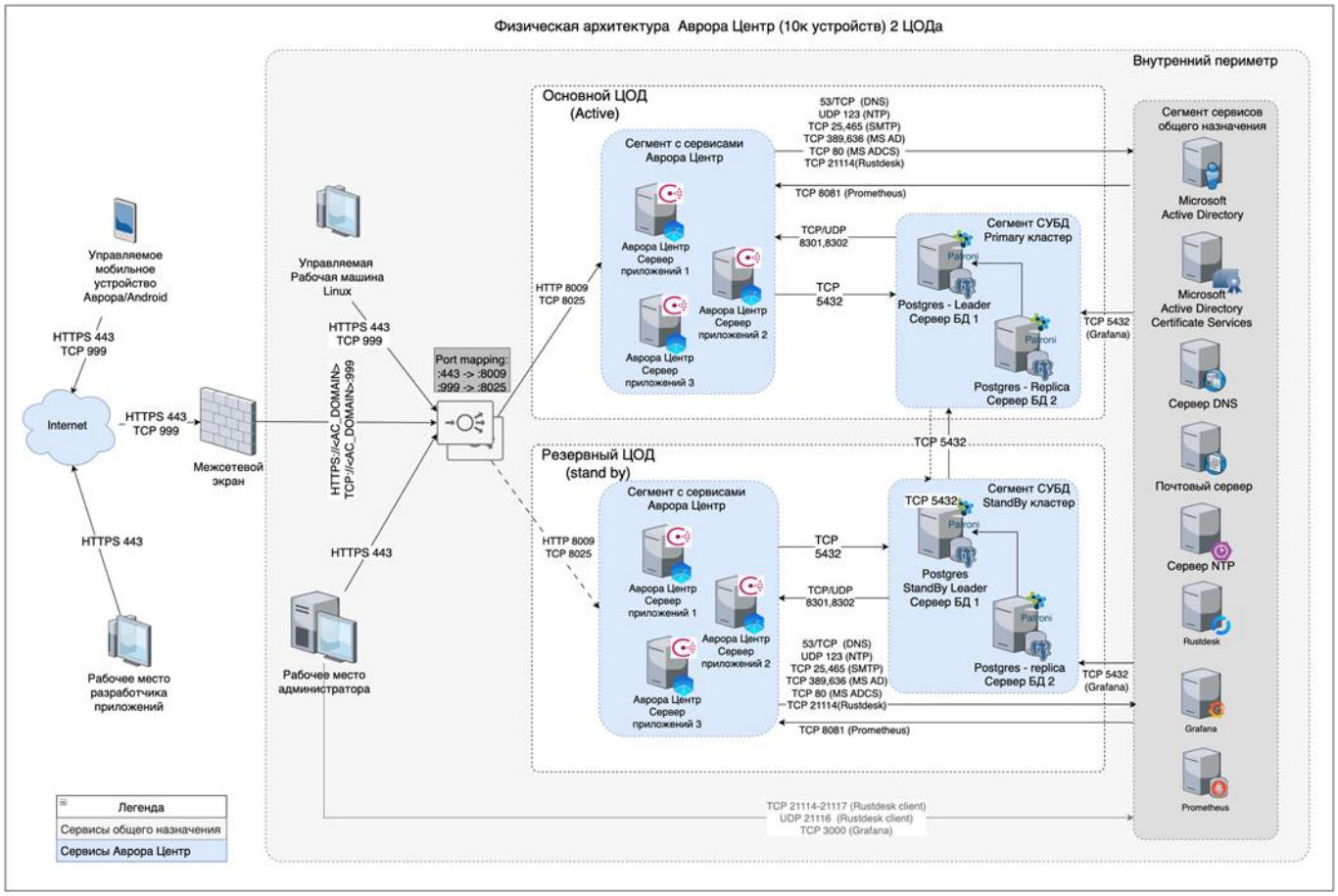


Рисунок 14

В каждом ЦОД устанавливается полный набор сервисов ППО в отказоустойчивой конфигурации (п. 2.3.4 – 2.3.6).

2.3.7.1. Серверы приложений и инфраструктурных компонентов

Приложения: сервисы приложений «Аврора Центр» активны в обоих ЦОД, но функционируют изолированно (межцодовое взаимодействие между ними отсутствует).

Инфраструктурные кластеры: в каждом ЦОД разворачиваются независимые, изолированные кластеры Redpanda, Valkey и Consul. Для синхронизациии файловых данных между серверами приложений настраивается сервер Syncthing и/или NFS.

2.3.7.2. Серверы баз данных (PostgreSQL)

Основной ЦОД функционирует в активном режиме (Active). СУБД функционирует в режиме Primary-кластер: один сервер выполняет роль Leader (Primary), второй сервер – роль локальной реплики (Replica), синхронизируя данные внутри этого же ЦОД. Переключение между Primary и локальной репликой в одном ЦОД происходит в автоматическом режиме. Резервный ЦОД функционирует в режиме ожидания (StandBy) – все сервисы активны, но трафик в данный ЦОД не проходит. СУБД функционирует в режиме Standby-кластер: один сервер выступает в роли Standby Leader и получает данные по схеме каскадной репликации из основного ЦОД, второй сервер является локальной репликой для Standby Leader.

2.3.7.3. Маршрутизация и балансировка трафика

Внешний балансировщик распределяет входящий трафик одновременно на обе площадки (Основную и Резервную). Правила балансировки:

– пользовательские сессии: направляются строго в один и тот же ЦОД для сохранения контекста (используются методы `IP Hash` или `Consistent Hashing`);

– запросы устройств: распределяются циклически (метод `Round-Robin`);

– длительные соединения для push-уведомлений: распределяются по методу `Least Connections`.

2.3.7.4. Сценарий аварии (отказ одного центра обработки данных)

Переключение Primary-сервера PostgreSQL на резервный ЦОД (`failover/switchover`) осуществляется в ручном режиме (п. 3.10.22). Сервисы Аврора Центр автоматически переподключаются к новому Primary-серверу PostgreSQL. Потеря данных при переключении БД будет равна задержке (лагу) репликации.

3. УСТАНОВКА ППО

ВНИМАНИЕ! Администратору/разработчику при копировании команд из настоящего документа в формате `.pdf` необходимо проявлять внимательность и дополнительно проверять результаты выполнения соответствующих команд на экране.

3.1. Общая информация

Установка ППО и компонентов среды функционирования ППО осуществляется с помощью сценариев установки ППО, выполняемых на управляющей ЭВМ и написанных с использованием декларативного языка разметки для описания конфигураций Ansible. Сценарии установки ППО позволяют выполнить установку как локально (все компоненты на 1 ЭВМ), так и с удаленной ЭВМ (управляющей ЭВМ) (Рисунок 15).

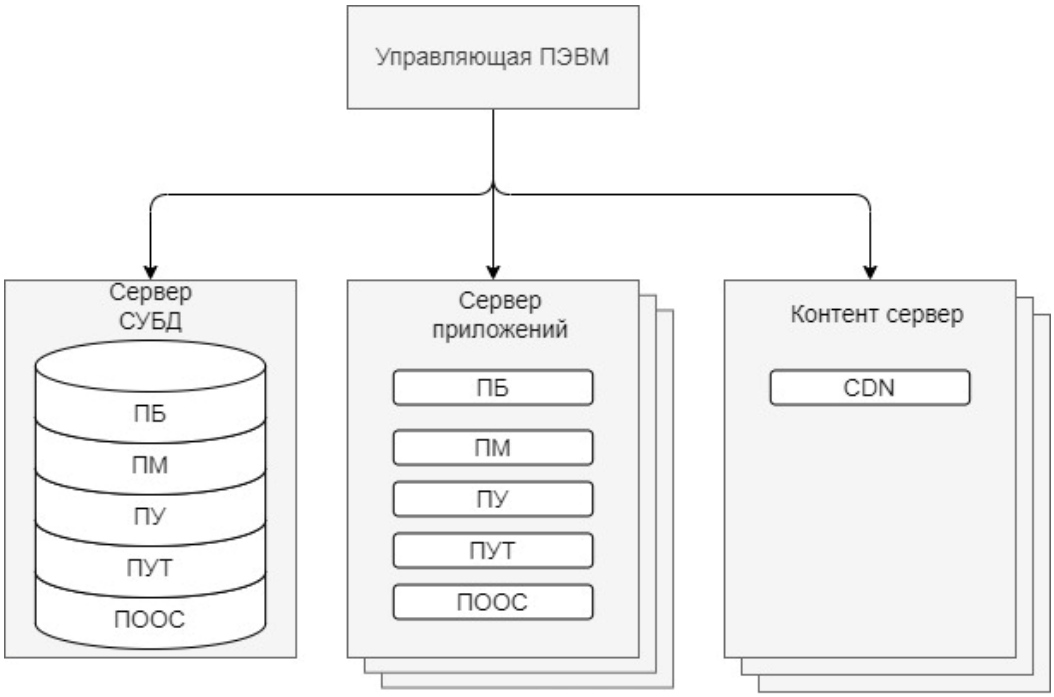


Рисунок 15

ПРИМЕЧАНИЕ. Управляющая ЭВМ необходима только для установки, настройки и управления ППО и не требуется для функционирования ППО.

ВНИМАНИЕ! Для установки ППО необходимо наличие стабильного интернет-соединения на серверах приложений, серверах БД, контент-серверах, а также на управляющей ЭВМ.

Для установки ППО необходимо выполнить следующие действия:

1) Убедиться, что соблюдены требования, приведенные в подразделе 1.4;

2) Установить и настроить ОС на серверы приложений, серверы БД и при необходимости на контент-серверы (подраздел 3.2);

ВНИМАНИЕ! Для корректной установки ППО на ОС Astra Linux Special Edition «Смоленск» необходимо отключить режим замкнутой программной среды (ЗПС). Режим ЗПС блокирует запуск бинарных файлов, не имеющих подписи производителя ОС Astra Linux. В этом режиме ОС препятствует работе компонентов ППО и его следует отключить, выполнив следующие действия:

– отключите режим ЗПС с помощью команды:

```
sudo astra-digsig-control disable
```

– обновите `initramfs`:

```
sudo update-initramfs -uk all
```

– перезапустите ОС:

```
sudo reboot
```

Далее следуйте рекомендациям, указанным в п. 3.6.1 и п. 3.6.2.

3) Развернуть и настроить управляющую ЭВМ (подраздел 3.3);

4) Настроить инфраструктурные компоненты ППО и компоненты среды функционирования ППО (п. 3.5.1);

5) Настроить ППО (п. 3.5.2);

6) Установить инфраструктурные компоненты ППО и компоненты среды функционирования ППО (п. 3.6.1);

7) Установить ППО (п. 3.6.2);

8) Настроить подсистемы ППО (подраздел 3.8);

9) При необходимости выполнить дополнительные настройки ППО, инфраструктурных компонентов ППО и среды функционирования ППО (подраздел 3.10);

10) Проверить корректность установки и функционирования ППО (подраздел 3.11).

3.2. Порядок установки и настройки ОС на серверах приложений, серверах БД и контент-серверах

3.2.1. Установить на серверы приложений, серверы БД и при необходимости на контент-серверы одну из следующих ОС, приведенных в п. 1.4.2.

ВНИМАНИЕ! Перед установкой ОС необходимо ознакомиться с требованиями, приведенными в документации на СЗИ НСД.

ОС должна быть установлена в **минимальной конфигурации без графического интерфейса**. Например, при установке ОС Альт необходимо выбрать конфигурацию «Minimal Install» (Рисунок 16) при установке ОС.



Рисунок 16

3.2.2. Обеспечить выполнение следующих требований

3.2.2.1. Требования к предустановленным в ОС пакетам

На серверах приложений, серверах БД и контент-серверах должны быть установлены следующие пакеты:

– `sudo`;

– `python` версии 3.6 или выше.

3.2.2.2. Требования к настройке сети ОС

Необходимо, чтобы настройки сети ОС соответствовали следующим требованиям:

1) Для основного сетевого интерфейса должен присутствовать конфигурационный файл(ы):

– ОС Альт:

```
/etc/net/ifaces/<имя интерфейса>/ipv4address
/etc/net/ifaces/<имя интерфейса>/ipv4route
/etc/net/ifaces/<имя интерфейса>/options
```

– ОС Astra Linux, Debian: `/etc/network/interfaces;`

– ОС Ubuntu: `/etc/netplan/*.yml;`

2) Сетевой интерфейс должен автоматически запускаться при загрузке ОС.

Для этого необходимо:

– в конфигурационном файле `/etc/sysconfig/network-scripts/ifcfg-<имя интерфейса>` (для ОС РЕД ОС) или `/etc/net/ifaces/<имя интерфейса>/options` (для ОС Альт) задать следующее значение параметра `ONBOOT`:

```
ONBOOT=yes
```

– в конфигурационный файл `/etc/network/interfaces` (для ОС Astra Linux или Debian) внести следующую запись:

```
auto <имя интерфейса>
```

– в ОС Ubuntu автозапуск сетевого интерфейса настроен по умолчанию.

3) Приоритеты в конфигурационном файле `/etc/nsswitch.conf` должны выглядеть следующим образом (при использовании `dnsmasq`):

```
hosts: files dns ...
```

где `...` – остальные опции, если они используются;

4) На сетевых интерфейсах серверов приложений, серверов БД и контент-серверов должны быть настроены статические ip-адреса (использование динамических адресов, выдаваемых по DHCP, не допускается).

5) В случае, когда сервера приложений ППО находятся за прокси-сервером, необходимо отключить проксирование запросов к сервисам ППО.

Для этого в переменной `no_proxy` конфигурационного файла `/etc/environment` необходимо указать список доменных имен или IP-адресов серверов приложений, для которых не следует использовать проксирование:

```
NO_PROXY=localhost,127.0.0.0/8,.local,<домен сервера приложений>
```

Например:

```
NO_PROXY=localhost,127.0.0.0/8,.local,omp.acenter.example
```

3.2.3. Перейти в учетную запись суперпользователя с помощью команды:

– ОС Альт и РЕД ОС:

```
su -
```

– ОС Astra Linux, ОС Debian и ОС Ubuntu:

```
sudo -i
```

3.2.4. Настроить репозитории:

– ОС Astra Linux SE версии 1.7:

В конфигурационном файле `/etc/apt/sources.list` необходимо исключить CD-ROM из списка доступных репозиториев, а также настроить доступ к основному (`main`) и базовому (`base`) репозиториям ОС:

```
#deb cdrom:[OS Astra Linux 1.7.9.41  1.7_x86-64 DVD ]/ 1.7_x86-64
contrib main non-free
deb https://download.astralinux.ru/astra/stable/1.7_x86-64/repository-
main/ 1.7_x86-64 main contrib non-free
deb https://download.astralinux.ru/astra/stable/1.7_x86-64/repository-
base/ 1.7_x86-64 main contrib non-free
deb https://download.astralinux.ru/astra/stable/1.7_x86-64/repository-
extended/ 1.7_x86-64 main contrib non-free
```

– ОС Astra Linux SE версии 1.8:

В конфигурационном файле `/etc/apt/sources.list` необходимо исключить CD-ROM из списка доступных репозиториев, а также настроить доступ к основному (`main`) и базовому (`base`) репозиториям ОС:

```
deb https://download.astralinux.ru/astra/stable/1.8_x86-64/repository-extended/ 1.8_x86-64 main contrib non-free non-free-firmware
deb https://download.astralinux.ru/astra/stable/1.8_x86-64/repository-main/ 1.8_x86-64 main contrib non-free non-free-firmware
#deb cdrom:[OS Astra Linux 1.8.4.48 1.8_x86-64 DVD]/ 1.8_x86-64 contrib main non-free non-free-firmware
```

– ОС Альт 8 СП релиз 10:

В конфигурационном файле `/etc/apt/sources.list.d/sources.list` необходимо исключить CD-ROM из списка доступных репозиториев, а также настроить доступ к основным (`main, classic`) репозиториям ОС:

```
#rpm cdrom:[ALT SP Server 11100-01 x86_64 build 2023-05-29]/ ALTLinux main

rpm http://ftp.altlinux.org/pub/distributions/ALTLinux p10/branch/x86_64 classic gostcrypto
rpm http://ftp.altlinux.org/pub/distributions/ALTLinux p10/branch/x86_64-i586 classic
rpm http://ftp.altlinux.org/pub/distributions/ALTLinux p10/branch/noarch classic
```

3.2.5. Назначить пользователям ОС права на выполнение команд от имени суперпользователя без ввода пароля с помощью команды:

```
echo '<имя пользователя> ALL=(ALL:ALL) NOPASSWD: ALL' | EDITOR='tee -a' visudo -f /etc/sudoers.d/<имя пользователя>
```

Например:

```
echo 'omp ALL=(ALL:ALL) NOPASSWD: ALL' | EDITOR='tee -a' visudo -f /etc/sudoers.d/omp
```

ВНИМАНИЕ! Права на выполнение команд от имени суперпользователя должны быть назначены всем пользователям (на управляющей ЭВМ, серверах приложений, серверах БД и контент-серверах), которыми осуществляется установка

компонентов среды функционирования, СУБД и ППО. В противном случае в процессе установки возникнут ошибки.

ПРИМЕЧАНИЕ. Пользователь «omp» используется для установки и обновления ППО. Данный пользователь может быть отключен в процессе работы ППО и включаться только при обновлении и настройке ППО.

3.2.6. Отключить настройку безопасности «astra-sudo-control» (в случае использования ОС Astra Linux):

```
sudo astra-sudo-control disable
```

3.2.7. Задать имя хоста с помощью команды:

```
hostnamectl set-hostname "имя_хоста.имя_домена"
```

ВНИМАНИЕ! При задании имени хоста обязательно должно быть задано имя домена, которое отделяется точкой. Например:

```
hostnamectl set-hostname ocs-app.local
```

3.2.8. В настройках DNS-сервера или файлах `/etc/hosts` указать имена хостов (`hostname`) и полные имена доменов (`FQDN`) всех серверов кластера:

```
"ip-адрес" "имя_хоста.имя_домена"
```

Например (в файле `/etc/hosts`):

```
192.168.0.108 ocs-app.local
```

3.2.9. Задать адреса DNS-серверов:

Адреса DNS-серверов задаются в файле `/etc/resolv.conf` в следующем формате:

```
nameserver "ip-адрес"
```

Например:

```
nameserver 192.168.0.1
```

ПРИМЕЧАНИЯ:

✓ В случае отсутствия файла `/etc/resolv.conf` необходимо его создать;

✓ При использовании ОС Ubuntu 24.04 вносить правки в файл `/etc/resolv.conf` не требуется.

3.2.10. Настроить маршрут по умолчанию (`default gateway`) через `lan` интерфейс в соответствии с документацией на ОС.

3.2.11. Задать текущие дату и время с помощью команды:

```
date -s 'YYYY-MM-DD HH:MI:SS'
```

Например:

```
date -s '2021-03-31 12:34:56'
```

3.2.12. Перезагрузить управляющую ЭВМ и серверы с помощью команды:

```
reboot
```

3.3. Порядок развертывания и настройки управляющей ЭВМ

ВНИМАНИЕ! Перед развертыванием и настройкой управляющей ЭВМ необходимо произвести установку, настройку ОС на серверах приложений, серверах БД и при необходимости на контент-серверах в соответствии с подразделом 3.2.

Для развертывания и настройки управляющей ЭВМ необходимо выполнить следующие действия:

3.3.1. Установить на управляющую ЭВМ одну из ОС, приведенных в таблице (Таблица 13).

Таблица 13

ОС	Версия
Альт Сервер	10.4
Альт 8 СП	10.2.2
Astra Linux Special Edition (Орел, Воронеж, Смоленск)	1.8.4.48
Debian	12.8
РЕД ОС	7.3.6
Ubuntu	22.04.5
Ubuntu	24.04.3

ПРИМЕЧАНИЕ. В качестве управляющей ЭВМ может использоваться как отдельная ЭВМ, так и сервер приложений ППО.

3.3.2. Настроить сетевое взаимодействие управляющей ЭВМ с серверами приложений, серверами БД и контент-серверами.

Настройка сети на управляющей ЭВМ осуществляется в соответствии с ЭД на ОС.

3.3.3. Настроить репозитории:

– ОС Astra Linux SE версии 1.8:

Для этого в конфигурационном файле `/etc/apt/sources.list` необходимо исключить CD-ROM из списка доступных репозиториев, а также настроить доступ к основному (`main`) и базовому (`base`) репозиториям ОС:

```
# deb cdrom:[OS Astra Linux 1.8.4.48 1.8_x86-64 DVD]/ 1.8_x86-64
contrib main non-free non-free-firmware
deb https://download.astralinux.ru/astra/stable/1.8_x86-64/repository-
extended/ 1.8_x86-64 main contrib non-free non-free-firmware
deb https://download.astralinux.ru/astra/stable/1.8_x86-64/repository-
main/ 1.8_x86-64 main contrib non-free non-free-firmware
```

– ОС Альт 8 СП релиз 10:

В конфигурационном файле `/etc/apt/sources.list.d/sources.list` необходимо исключить CD-ROM из списка доступных репозиториев, а также настроить доступ к основным (`main, classic`) репозиториям ОС:

```
#rpm cdrom:[ALT SP Server 11100-01 x86_64 build 2023-05-29]/ ALTLinux
main

rpm http://ftp.altlinux.org/pub/distributions/ALTLinux
p10/branch/x86_64 classic gostcrypto
rpm http://ftp.altlinux.org/pub/distributions/ALTLinux
p10/branch/x86_64-i586 classic
rpm http://ftp.altlinux.org/pub/distributions/ALTLinux
p10/branch/noarch classic
```

3.3.4. Создать на управляющей ЭВМ отдельный каталог, скопировать в него содержимое каждого DVD, содержащего ППО, инфраструктурные компоненты и требуемые клиентские приложения, затем перейти в созданный каталог с помощью команды:

```
cd <путь к каталогу>
```

В результате должна получиться следующая структура:

```
/
├── AC_5.7.0_installer-ac-mt.sh
├── AC_5.7.0_install-ac-mt.tar.gz
├── AC_5.7.0_install-infra.tar.gz
└── AC_5.7.0_<client-apps>.tar.gz
```

Например:

```
/
├── AC_5.7.0_installer-ac-mt.sh
├── AC_5.7.0_install-ac-mt.tar.gz
├── AC_5.7.0_install-infra.tar.gz
├── AC_5.7.0_client-apps-aurora.tar.gz
├── AC_5.7.0_client-apps-android.tar.gz
├── AC_5.7.0_client-apps-linux-alt.tar.gz
├── AC_5.7.0_client-apps-linux-astra.tar.gz
├── AC_5.7.0_client-apps-linux-redos.tar.gz
└── AC_5.7.0_client-apps-linux-ubuntu.tar.gz
```

3.3.5. Запустить AC_<версия ППО>_installer-ac-mt.sh с помощью команды:

```
bash AC_<версия ППО>_installer-ac-mt.sh
```

3.3.6. Перейти в каталог со сценариями установки с помощью команды:

```
cd install-<версия ППО>/install-ac-mt/
```

Например:

```
cd install-release-v5.3.0/install-ac-mt/
```

ПРИМЕЧАНИЕ. Дальнейшие действия по установке и настройке компонентов среды функционирования ППО, а также ППО, необходимо выполнять из данного каталога.

3.3.7. Установить на управляющей ЭВМ пакеты, необходимые для запуска сценариев установки:

– установить системные пакеты с помощью команды:

```
sudo bash ./cnp-system-pkgs.sh
```

– установить PyPI-пакеты с помощью команды:

```
bash ./cnp-python-pkgs.sh
```

ПРИМЕЧАНИЕ. Скрипт установки PyPI-пакетов может выдать предупреждение об отсутствии каталога с исполняемыми файлами в переменной PATH. Данное предупреждение указывает на то, что установленные Python-пакеты могут оказаться недоступными для вызова из командной строки.

Рекомендуется добавить в переменную PATH каталог с установленными пакетами в соответствии с рекомендацией.

ПРИМЕЧАНИЕ. Если требуется использовать собственный репозиторий PyPI-пакетов (`PyPI registry`), то предварительно необходимо создать конфигурационный файл `/etc/pip.conf` содержащий следующие настройки:

```
[global]
index-url = <адрес репозитория PyPI>
trusted-host = <домен хоста с репозиторием>
```

Например:

```
[global]
index-url = https://example.com/repository/pypi-pypi.org/simple
trusted-host = example.com
```

ПРИМЕЧАНИЕ. В Ansible-core 2.18 и выше прекращена поддержка Python 3.7. Таким образом, для работы с управляемыми ЭВМ на ОС с устаревшей версией Python (таких как ОС Astra Linux 1.7) необходимо принудительное ограничение версии Ansible-core.

```
ANSIBLE_CORE_VERSION="<2.18" cnp-python-pkgs.sh
```

3.3.8. Настроить SSH доступ управляющей ЭВМ к серверам приложений, серверам БД и контент-серверам (даже в случае, когда управляющая ЭВМ и серверы установлены на 1 ЭВМ):

– сформировать ключевую пару на управляющем сервере:

```
ssh-keygen -t rsa -b 4096
```

– скопировать открытый ключ на серверы приложений, серверы БД и контент-серверы:

```
ssh-copy-id <имя пользователя>@<сервер приложений>
ssh-copy-id <имя пользователя>@<сервер БД>
ssh-copy-id <имя пользователя>@<контент сервер>
```

– проверить доступ с управляющей машины на серверы приложений, серверы БД и контент-серверы по SSH ключу (при выполнении команд ниже ввод пароля не должен требоваться):

```
ssh <имя пользователя>@<сервер приложения>
ssh <имя пользователя>@<сервер БД>
ssh <имя пользователя>@<контент сервер>
```

ПРИМЕЧАНИЕ. Управляющие команды, формируемые сценариями установки ППО, передаются с использованием протокола SSH.

3.4. Упрощенная настройка ППО с помощью интерактивного меню

Упрощенная настройка используется для быстрой настройки базовой конфигурации ППО. Настройка ППО сводится к указанию в интерактивном меню только самых необходимых параметров: имя хоста сервера приложений ППО, имя хоста СУБД, адрес внешнего балансировщика, версия СУБД PostgreSQL.

Упрощенная установка позволяет установить ППО в конфигурациях:

– все в одном (ППО и СУБД на одном сервере);

– один сервер приложений и один сервер БД.

Данный вариант рекомендуется использовать в ознакомительных целях с базовыми возможностями ППО, в остальных случаях следует пропустить данный пункт.

В процессе ознакомительной установки будут сформированы инвентарный файл окружения (файл: `inventories/<название окружения>/hosts.yml`), а также конфигурационные файлы окружения (каталог: `config/environments/<название окружения>/`). Подробная информация о конфигурационных файлах окружений приведена в п. 13.2.8. При необходимости изменения в данные файлы можно внести вручную.

Для ознакомительной установки ППО необходимо выполнить следующие действия:

3.4.1. Запустить сценарий установки с помощью команды:

```
ANSIBLE_USER="<имя пользователя>" ./deploy.sh -e <название окружения>
```

ПРИМЕЧАНИЕ. Название окружения может иметь любое значение.

Например:

```
ANSIBLE_USER="omp" ./deploy.sh -e test_env
```

3.4.2. В процесс выполнения скрипта необходимо в интерактивном режиме задать следующие параметры:

– имя хоста сервера приложений ППО (Set Aurora Center APP host), например:

```
Set Aurora Center APP host: ocs-app.local
```

– имя хоста сервера БД, например:

```
Set Aurora Center DB host: ocs-app.local
```

– URL-адрес внешнего балансировщика (при его наличии), например:

```
Use external balancer [n]: y
Set Aurora Center external URL(example - http(s)://mydomain.com):
http://mydomain.com
```

ВНИМАНИЕ! URL-адрес указывается без символа «/» в конце.

– версию СУБД PostgreSQL, например:

```
PostgreSQL version [15]: 15
```

3.4.3. Подтвердить корректность заданных параметров, например:

```
Ansible deployment user             = omp
Aurora Center APP host              = ocs-app.local
Aurora Center DB host               = ocs-app.local
Aurora Center external URL          = http://mydomain.com
PostgreSQL version                  = 15

is this ok [y/n]? y
```

В случае подтверждения (y), начнется установка ППО. В случае отказа (n), будет выполнено завершение сценария установки. Для продолжения работы перейти к подразделу 3.7.

3.5. Порядок настройки компонентов среды функционирования ППО и ППО

ПРИМЕЧАНИЕ. Сценарии установки позволяют выполнить настройку и установку ППО, а также компонентов среды функционирования ППО для нескольких различных окружений. Порядок конфигурирования и установки ППО для нескольких окружений приведен в п. 3.10.16.

3.5.1. Настройка компонентов среды функционирования ППО и инфраструктурных компонентов ППО

ПРИМЕЧАНИЕ. Пароли, секреты, токены должны быть не менее 8 символов, а также содержать заглавные и строчные буквы (кириллица не допускается), цифры, пробелы и специальные символы. Допускается использовать следующие специальные символы:

```
$&*()-=_;.
```

Для настройки компонентов среды функционирования ППО и инфраструктурных компонентов ППО необходимо выполнить следующие действия:

3.5.1.1. Перейти в каталог со сценариями установки `/install-<версия ППО>/install-ac-mt/`.

3.5.1.2. В инвентарном файле `inventories/hosts.yml` задать адреса серверов (имена хостов), на которые будут установлены компоненты среды функционирования ППО и инфраструктурные компоненты ППО.

Описание порядка задания адресов в инвентарном файле `inventories/hosts.yml` приведено в п. 3.10.13.

Для отображения адреса ЭВМ необходимо выполнить команду:

```
hostname
```

Примеры файлов `hosts.yml` для однонодовой и кластерной конфигурации приведены в каталоге `samples/ac/inventories/`.

Описание параметров инвентарного файла `inventories/hosts.yml` приведено в п. 12.1.1.

3.5.1.3. В конфигурационном файле `config/vars/_vars.yml` необходимо задать либо поменять предустановленные значения следующих параметров:

– параметры подключения подсистем ППО к БД:

```
postgresql:
  port: 5432
```

При использовании балансировщика БД необходимо задать адрес хоста балансировщика, например:

```
postgresql:
  host: "10.189.221.57"
```

– пароль суперпользователя "`postgres`" СУБД Postgresql, если установка СУБД осуществляется с помощью сценариев установки:

```
pg_superuser_password: "postgres"
```

– версию СУБД:

```
pg_version: 15
```

Перечень допустимых значений параметра приведен в таблице (Таблица 14).

Таблица 14

Значение параметра	Версия СУБД
14	PostgreSQL 14
15	PostgreSQL 15
16	PostgreSQL 16
17	PostgreSQL 17
14-pro	Postgres Pro Standard 14
14-stdcert	Postgres Pro Certified 14
15-pro	Postgres Pro Standard 15
15-stdcert	Postgres Pro Certified 15
16-pro	Postgres Pro Standard 16
16-stdcert	Postgres Pro Certified 16
17-pro	Postgres Pro Standard 17
17-stdcert	Postgres Pro Certified 17

– имя и пароль пользователя СУБД Postgresql с ролью «replication»:

```
pg_replication_user:
  name: replication
  password: 123FD5648ert**h
```

– имя и пароль суперпользователя СУБД Postgresql, от имени которого будет осуществляться установка ППО:

```
pg_custom_superuser:
  username: ocs_superuser
  password: ClacVob^Twes0Ls6
```

– адреса DNS-серверов (обязательно для ОС Ubuntu 22.04, Ubuntu 24.04 и РЕД ОС 8.0), например:

```
dnsmasq_upstream_servers: "192.168.137.1,10.189.211.10"
```

В случае нескольких адресов на сетевом интерфейсе, необходимо указать адрес, на котором запускать инфраструктурные компоненты, явно, через переменные:

```
consul_ip: "<ip-адрес>"
valkey_ip: "<ip-адрес>"
```

ПРИМЕЧАНИЕ. Описание параметров конфигурационных файлов сценариев установки среды функционирования ППО, сценариев установки ППО и ППО приведено в самих конфигурационных файлах в виде комментариев.

3.5.1.4. В конфигурационном файле `config/secret.yml` задать пароли, секреты и токены:

– пароль доступа к БД:

```
database:
  password: CHANGEME-ocs
```

– пароль доступа к СУБД Valkey в параметре `valkey_password`:

```
valkey:
  password: "CHANGEME-@rTT9089087fslk"
```

– секретный ключ для аутентификации запросов к сервисам ППО:

```
hmac:
  key: "CHANGEME-DEFAULT-F1IWp0t5dY5lYJrm7H-DEFAULT"
```

– токен доступа к системе обнаружения сервисов `Consul`:

```
consul:
  token: "CHANGEME-ae9f5abb-6b8f-9252-59c5-53bcb651f182"
```

– секретный ключ клиентов (сервисов):

```
defaultOidcClientSecret: "CHANGEME-HWfwehfoIOHwfe233WEfvwewe"
```

– ключ шифрования секретов сервиса `ocs-auth-config-api`, хранящихся в БД:

```
encrypt:
  keys:
    - CHANGEME-master-key
    - master-key-example
```

– ключ шифрования секретов сервиса `ocs-emm-policies-api`, хранящихся в БД:

```
encryption:
  keys:
    - CHANGEME-master-key
    - master-key-example
```

– ключ шифрования секретов сервиса `ocs-appstore-settings-api`, хранящихся в БД:

```
appstoreSecrets:
  settingsApiSecrets:
    - "you really need to change this"
```

ВНИМАНИЕ! При обновлении ППО удалять старые ключи запрещается. Новые ключи необходимо добавлять в начало списка;

– пароли, используемые для защиты критичной информации (например, cookie сессии):

```
oidcpSecrets:
  system:
    - kdj%93cxk+57nMa4
  cookie:
    - 9v_wer8*&r=_hY8u
```

ВНИМАНИЕ! Длина пароля должна быть не менее 16 символов. При обновлении ППО удалять старые пароли запрещается. Новые пароли необходимо добавлять в начало списка;

– пароль доступа к сервису гарантированной доставки сообщений `Redpanda`:

```
redpanda:
  password: "CHANGEME-%GJJ690t5-0"
```

– пароль суперпользователя сервиса гарантированной доставки сообщений `Redpanda`:

```
redpanda_superuser:
  name: admin
  password: "CHANGEME-Tes%3@@poi"
```

– пароль доступа к сервису управления кластером БД `Patroni`:

```
patroni:
  api_user: 'patroni'
  api_password: "CHANGEME-VV4445@@@3kjj"
```

– пароль доступа к сервису балансировки нагрузки Keepalived:

```
keepalived_auth_pass: "CHANGEME-ravJulis*Im5"
```

3.5.2. Настройка ППО (подсистем ППО)

ВНИМАНИЕ! Перед выполнением настроек необходимо изучить порядок работы с конфигурационными файлами, приведенный в п. 13.2.9.

ПРИМЕЧАНИЕ. Пароли, секреты, токены должны быть не менее 8 символов, а также содержать заглавные и строчные буквы (кириллица не допускается), цифры, пробелы и специальные символы. Допускается использовать следующие специальные символы:

```
$&*()-=_;.
```

Для настройки ППО необходимо выполнить следующие действия:

3.5.2.1. В инвентарном файле inventories/hosts.yml задать адреса серверов (имена хостов), на которые будут установлены подсистемы ППО.

Описание порядка задания адресов в инвентарном файле inventories/hosts.yml приведено в п. 3.10.13.

3.5.2.2. Выполнить настройку порта для административных (привилегированных) интерфейсов ППО (при необходимости).

По умолчанию привилегированные и непривилегированные интерфейсы принимают запросы на порту 8009.

В ППО предусмотрена возможность назначить административным (привилегированным) интерфейсам ППО отдельный порт. Это позволяет ограничить доступ непривилегированных пользователей к административным (привилегированным) интерфейсам ППО.

Для назначения отдельного порта для административного интерфейса необходимо в конфигурационном файле `config/vars/_vars.yml` указать порты для непривилегированных интерфейсов (параметр: `nginx_vhost_external_port`) и административных (привилегированных) интерфейсов (параметр: `nginx_vhost_external_admin_port`), например:

```
nginx_vhost_external_port: 8009
nginx_vhost_external_admin_port: 8010
```

Изменение порта необходимо учитывать в настройках URI в пп. 3.5.2.3.

3.5.2.3. Выполнить настройку TLS для внешних интерфейсов ППО (при необходимости).

Настройка TLS может потребоваться для защиты каналов связи, например, между внешним балансировщиком и сервером приложений или между управляемыми устройствами и сервером приложений.

По умолчанию сервер принимает запросы по протоколу HTTP. Для настройки сервера на обработку запросов по защищенному протоколу TLS необходимо выполнить следующие действия:

3.5.2.3.1 В конфигурационном файле `config/vars/_vars.yml` включить опцию `nginx_vhost_external_tls` и указать закрытый ключ и цепочку сертификата закрытого ключа в PEM формате, например:

```
nginx_vhost_external_tls:
  enabled: true
  private_key: |
    -----BEGIN PRIVATE KEY-----
    ...
    -----END PRIVATE KEY-----
  certificate: |
    -----BEGIN CERTIFICATE-----
    ...
    -----END CERTIFICATE-----
```

ПРИМЕЧАНИЕ. Сертификат закрытого ключа должен соответствовать следующим требованиям:

– в поле `Common Name (CN)` должно быть указано доменное имя;

– в сертификате должно присутствовать поле `subjectAltName`, значение которого должно совпадать со значением в поле `Common Name (CN)`;

– сертификат должен входить в цепочку доверия сертификатов на устройствах;

– файл сертификата закрытого ключа должен включать цепочку сертификатов.

Пример создания закрытого ключа и сертификата закрытого ключа приведен в пп. 3.8.1.2.

3.5.2.3.2 В конфигурационном файле `config/vars/_vars.yml` необходимо отделить порт для внешних вызовов (параметр: `nginx_vhost_external_port`) от внутреннего порта для межподсистемного взаимодействия (параметр: `nginx_vhost_system_port`). Например, указать следующие порты:

```
nginx_vhost_external_port: 8009
nginx_vhost_system_port: 8007
```

ПРИМЕЧАНИЕ. Настройки TLS соединения будут применены к следующими портам:

– порт для внешних вызовов (параметр: `nginx_vhost_external_port`);

– порт административного интерфейса (параметр: `nginx_vhost_external_admin_port`);

– порт для приема запросов от контент-серверов (параметр: `nginx_vhost_external_cdn_origin_port`).

ПРИМЕЧАНИЕ. Для контейнерной установки (All-in-One) при запуске ППО из единого docker-образа по протоколу HTTPS необходимо убедиться, что выпущенные сертификаты и ключи переданы во внутреннюю среду контейнера. При запуске необходимо использовать параметры монтирования, указывающие на локальные файлы, например:

```
--volume "$(pwd)/ac570.local.key:/etc/consul-template/ssl.key"
          --volume "$(pwd)/ac570.local.chain.crt:/etc/consul-
template/ssl.crt"
```

Инструкция по созданию самоподписанных сертификатов приведена в пп. 3.8.1.2.

3.5.2.4. Выполнить настройку URL-адресов ППО.

URL-адреса ППО задаются в секции `publicUris` в конфигурационном файле `config/config.yml.j2`.

ВНИМАНИЕ! URL-адреса указываются без символа «/» в конце.

По умолчанию URL-адреса ППО настроены следующим образом:

– протокол: `HTTP`;

– `hostname`: соответствует первой записи в группе `app` в инвентарном файле `inventories/hosts.yml`;

– порт: соответствует переменной `nginx_vhost_external_port`, заданной в конфигурационном файле `config/vars/_vars.yml`.

Данные настройки соответствуют однонодовой конфигурации системы с незащищенным соединением, общим портом для привилегированных и непривилегированных интерфейсов и без использования внешнего балансировщика.

Возможны следующие варианты настройки:

3.5.2.4.1 URL-адреса ППО используют 1 домен, без внешнего балансировщика.

Данные настройки применимы только к однонодовой конфигурации. В параметре `publicUris.ac.commonAddress` – по умолчанию указан `hostname`, соответствующий первой записи в группе `app` в инвентарном файле `inventories/hosts.yml`

```
    ac:
      commonAddress:
"http://{{groups['app']|first}}:${nginxVhostExternalPort}"
```

Если в пп. 3.5.2.2 был назначен отдельный порт для административных (привилегированных) интерфейсов, то необходимо в переменной `publicUris.ac.adminAddress` указать этот порт. Для указания порта можно использовать переменную `nginxVhostExternalAdminPort` или непосредственно задать значение:

```
    ac:
      adminAddress:
"http://{{groups['app']|first}}:${nginxVhostExternalAdminPort}"
```

3.5.2.4.2 Настройка URL-адресов ППО при использовании защищенного соединения.

Незащищенное (протокол HTTP) соединение с сервером приложений ППО допустимо использовать в пилотных проектах, где отсутствует обработка конфиденциальной информации. В остальных случаях должна обеспечиваться защита каналов связи.

ППО поддерживает возможность использования защищенного соединения только с использованием внешнего криптошлюза или настройки TLS на внешнем балансировщике.

При использовании защищенного соединения в URL-адресах ППО необходимо указывать протокол HTTPS:

```
    ac:
      commonAddress: "https://acenter.example.ru"
```

3.5.2.4.3 URL-адреса ППО используют 1 домен на внешнем балансировщике.

В данном случае в параметре `commonAddress` необходимо задать протокол (http или https), имя домена <AC_DOMAIN> и порт, настроенные на внешнем балансировщике, например:

```
    ac:
      commonAddress: "https://acenter.example.ru"
```

3.5.2.4.4 URL-адреса ППО разделены на внешний и внутренний домены на внешнем балансировщике.

Подобное разделение, требуется, например, когда необходимо ограничить доступ к Консолям администраторов из внешней сети.

В данном случае в параметре `commonAddress` необходимо задать протокол (http или https), имя домена <AC_DOMAIN> и порт для внешних адресов, а в параметре `adminAddress` задать протокол, имя домена и порт для внутренних адресов, например:

```
ac:
  commonAddress: "https://acenter.example.ru"
  adminAddress: "https://admin.example
```

3.5.2.4.5 URL-адреса ППО используют разные домены.

В данном случае для каждой подсистемы ППО задается свой собственный домен, например:

```
cdn:
  address: "https://external.cdn.example.ru"
  originAddress: "https://acenter.example.ru:8024"
auth:
  adminCrossTenantAddress: "https://authadmin.example"
  publicAddress: "https://authpublic.acenter.example.ru"
aps:
  adminAddress: "https://apsadmin.acenter.example.ru"
  devAddress: "https://apsdev.acenter.example.ru"
  marketAddress: "https://apsmarket.acenter.example.ru"
push:
  adminAddress: "https://pushadmin.example"
  publicAddress: "https://pushpublic.acenter.example.ru"
mt:
  adminAddress: "https://mt.example"
pkgrepo:
  adminAddress: "https://pkgrepoadmin.example"
  mobileAddress: "https://pkgrepomobile.acenter.example.ru"
  repoAddress: "https://pkgrepo.acenter.example.ru"
```

3.5.2.5. Отредактировать конфигурационный файл `config/config.yml.j2`.

В данном конфигурационном файле необходимо задать либо изменить предустановленные значения:

– домен учетных записей пользователей для тенанта `"default"`:

```
defaultIdentityDomain: "omprussia.ru"
```

– уровень детализации сообщений логирования (рекомендуется задать "info" при тестовой эксплуатации и "warn" при промышленной эксплуатации):

```
  logger:
    level: "info"
```

3.5.2.6. Настроить файловое хранилище ППО.

Описание настройки файлового хранилища ППО (файловых хранилищ ПМ, ПУ и ПООС) приведено в подразделе 3.9.

3.5.2.7. Выполнить настройки безопасности ППО, другие дополнительные настройки ППО и настройки подсистем ППО (при необходимости).

ВНИМАНИЕ! Перед установкой ППО требуется выполнить настройки безопасности ППО, дополнительные настройки ППО и настройки подсистем ППО (при необходимости).

Перечень и описание дополнительных настроек ППО приведен в подразделе 3.10.

3.6. Порядок установки компонентов среды функционирования ППО и ППО

3.6.1. Установка компонентов среды функционирования ППО и инфраструктурных компонентов ППО

3.6.1.1. Обеспечить синхронизацию времени между нодами кластера.

При эксплуатации ППО в кластерной конфигурации необходимо обеспечить синхронизацию времени между нодами кластера (например, с помощью утилиты chrony).

Для проверки синхронизации времени необходимо выполнить команду:

```
ansible-playbook play-check-time-on-hosts.yml --inventory-file
inventories/hosts.yml -vv --diff
```

По результатам выполнения команды будет выведено текущее время на каждой ноде кластера, например:

```
acenterapp03.ompcloud 2022-11-09 09:48:38.394115
acenterapp04.ompcloud 2022-11-09 09:48:38.394533
acenterapp05.ompcloud 2022-11-09 09:48:38.394939
acenterapp06.ompcloud 2022-11-09 09:48:38.395490
acenterapp01.ompcloud 2022-11-09 09:48:38.393034
acenterapp02.ompcloud 2022-11-09 09:48:38.393631
```

3.6.1.2. Установить на серверы приложений, серверы БД и контент-серверы необходимые пакеты.

ВНИМАНИЕ! После завершения установки пакетов службы SELinux и Firewalld будут отключены.

Для установки пакетов необходимо выполнить следующие действия:

3.6.1.2.1 Установить пакеты с помощью следующей команды:

```
ansible-playbook -i inventories/hosts.yml play-managed-node-
prerequisites.yml -vv -u <имя пользователя>
```

Также предусмотрена возможность установки пакетов вместе с компонентами среды функционирования ППО и инфраструктурными компонентами ППО с помощью команды:

```
ANSIBLE_USER="<имя пользователя>" ./deploy-infra.sh --install-
prerequisites
```

Например:

```
ANSIBLE_USER="omp" ./deploy-infra.sh --install-prerequisites
```

Для отдельной установки пакетов необходимо выполнить следующие команды:

– серверы приложений:

```
ansible-playbook -i inventories/hosts.yml play-managed-node-
prerequisites.yml -vv -u <имя пользователя> --extra-vars
"node_type=app" --limit app
```

– серверы БД:

```
ansible-playbook -i inventories/hosts.yml play-managed-node-
prerequisites.yml -vv -u <имя пользователя> --extra-vars
"node_type=db" --limit postgresql
```

– контент-серверы:

```
ansible-playbook -i inventories/hosts.yml play-managed-node-
prerequisites.yml -vv -u <имя пользователя> --extra-vars
"node_type=content" --limit content
```

– сервер с инфраструктурными компонентами (в случае, когда инфраструктурные компоненты устанавливаются на отдельный сервер):

```
ansible-playbook -i inventories/hosts.yml play-managed-node-
prerequisites.yml -vv -u <имя пользователя> --extra-vars
"node_type=app" --limit=consul,redpanda,valkey
```

Например, установка пакетов на серверы приложений осуществляется с помощью команды:

```
ansible-playbook -i inventories/hosts.yml play-managed-node-
prerequisites.yml -vv -u omp --extra-vars "node_type=app" --limit app
```

3.6.1.2.2 На серверах приложений, серверах БД и контент-серверах под управлением ОС РЕД ОС включить автозапуск службы `network` с помощью команды:

```
sudo systemctl enable network
```

3.6.1.2.3 Перезагрузить серверы приложений, серверы БД и контент-серверы с помощью команды:

```
sudo reboot
```

Порядок действий для самостоятельной установки пакетов, а также отключению служб SELinux и Firewalld приведен в п. 3.10.8 и 3.10.9.

3.6.1.3. Перейти в каталог со сценариями установки в соответствии с п. 3.3.6.

3.6.1.4. Установить компоненты среды функционирования ППО и инфраструктурные компоненты ППО с помощью команды:

```
ANSIBLE_USER="<имя пользователя>" ./deploy-infra.sh
```

Например:

```
ANSIBLE_USER="omp" ./deploy-infra.sh
```

В результате выполнения команды в каталоге `logs` будет сформирован лог-файл установки компонентов среды функционирования ППО и инфраструктурных компонентов ППО.

Описание параметров запуска скрипта `deploy-infra.sh` и их возможные значения приведены в подразделе 4.1.

ВНИМАНИЕ! Скрипт `deploy-infra.sh` позволяет устанавливать только СУБД PostgreSQL 14/15/16/17. СУБД Postgres Pro необходимо устанавливать самостоятельно.

При использовании СУБД Postgres Pro либо, если установку СУБД PostgreSQL 14/15/16/17 необходимо выполнить самостоятельно (без использования сценариев установки компонентов среды функционирования ППО и инфраструктурных компонентов ППО), команда установки компонентов среды функционирования ППО и инфраструктурных компонентов ППО имеет следующий вид:

```
ANSIBLE_USER="<имя пользователя>" ./deploy-infra.sh --skip-database
```

Описание установки и настройки СУБД Postgres Pro, а также требования к самостоятельной установке СУБД приведены в подразделе 3.12.

Также предусмотрена возможность установки компонентов среды функционирования ППО и инфраструктурных компонентов ППО по отдельности с помощью следующих команд:

```
ANSIBLE_USER="<имя пользователя>" ./deploy-infra.sh -c dnsmasq
ANSIBLE_USER="<имя пользователя>" ./deploy-infra.sh -c nginx
ANSIBLE_USER="<имя пользователя>" ./deploy-infra.sh -c consul
ANSIBLE_USER="<имя пользователя>" ./deploy-infra.sh -c consul-content
ANSIBLE_USER="<имя пользователя>" ./deploy-infra.sh -c consul-template
ANSIBLE_USER="<имя пользователя>" ./deploy-infra.sh -c redpanda
ANSIBLE_USER="<имя пользователя>" ./deploy-infra.sh -c valkey
ANSIBLE_USER="<имя пользователя>" ./deploy-infra.sh -c ocs-user
ANSIBLE_USER="<имя пользователя>" ./deploy-infra.sh -c db
```

3.6.2. Установка ППО

Для установки ППО необходимо выполнить команду:

```
ANSIBLE_USER="<имя пользователя>" ./deploy-ac.sh
```

Описание параметров запуска скрипта `deploy-ac.sh` и их возможные значения приведены в подразделе 4.2.

Например:

```
ANSIBLE_USER="omp" ./deploy-ac.sh
```

В результате выполнения команды в каталоге `logs` будет сформирован лог-файл установки ППО.

Для установки подсистем по отдельности необходимо в параметре `--subsystems` задать имя подсистемы.

ВНИМАНИЕ! Установка подсистем ППО должна осуществляться строго в следующей последовательности: ПБ, ПМ, ПООС, ПУ, ПУТ, CDN, ПСУ.

Пример установки подсистем по отдельности:

```
ANSIBLE_USER="omp" ./deploy-ac.sh --subsystems auth
ANSIBLE_USER="omp" ./deploy-ac.sh --subsystems appstore
ANSIBLE_USER="omp" ./deploy-ac.sh --subsystems pkgrepo
ANSIBLE_USER="omp" ./deploy-ac.sh --subsystems emm
ANSIBLE_USER="omp" ./deploy-ac.sh --subsystems mt
ANSIBLE_USER="omp" ./deploy-ac.sh --subsystems cdn
ANSIBLE_USER="omp" ./deploy-ac.sh --subsystems push
```

Если необходимо установить ПСУ отдельно от других подсистем ППО, тогда достаточно установить ПБ и ПСУ с помощью команды:

```
ANSIBLE_USER="<имя пользователя>" ./deploy-ac.sh --subsystems
auth,push
```

Для установки ППО без ПСУ необходимо выполнить команду:

```
ANSIBLE_USER="<имя пользователя>" ./deploy-ac.sh --subsystems
auth,appstore,pkgrepo,emm,mt
```

3.6.3. Выполнение настройки подсистем ППО

ВНИМАНИЕ! При невыполнении данных настроек часть функции ППО может не работать или работать некорректно.

Настройка подсистем ППО осуществляется в соответствии с подразделом 3.8.

3.6.4. Выполнение ограничений по применению

При эксплуатации ППО необходимо соблюдать следующие ОГРАНИЧЕНИЯ ПО ПРИМЕНЕНИЮ:

– ПСУ не осуществляет аутентификацию подключаемых к нему устройств под управлением ОС Аврора версии 5.0.0 и ниже, а также ОС Android, поэтому при необходимости обеспечения конфиденциальности, целостности и доступности push-уведомлений необходимо использовать компенсирующие меры защиты информации, например, криптографическую защиту канала связи с двусторонней аутентификацией между Сервером приложений ПСУ и устройствами. Для устройств под управлением ОС Аврора версии 5.1 и выше в ПСУ реализована аутентификация, поэтому использование компенсирующих мер не требуется;

– после установки и настройки ППО необходимо выполнить ограничения по применению, произвести настройки безопасности компонентов среды функционирования ППО, инфраструктурных компонентов ППО и настроить СЗИ. Необходимая информация приведена в п. 3.10.5.

3.6.5. Проверка корректности установки и функционирования ППО

Проверка осуществляется в соответствии с подразделом 3.11.

3.7. Адреса веб-консолей

Первоначальный вход в ППО осуществляется с помощью Консоли администратора ПБ и предустановленной учетной записи с ролью Администратор учетных записей:

– логин: admin@omprussia.ru;

– пароль: Admin123!

ПРИМЕЧАНИЕ. При первом входе в ППО необходимо сменить пароль.

В таблице (Таблица 15) приведены адреса веб-консолей.

Таблица 15

Веб-консоль	URL-адрес веб-консоли
Консоль администратора ПБ	http(s)://<сервер приложения>:8009/auth/admin/ui
Консоль администратора ПМ	http(s)://<сервер приложения>:8009/appstore/admin/ui
Консоль разработчика ПМ	http(s)://<сервер приложения>:8009/appstore/dev/ui
Консоль администратора ПУ	http(s)://<сервер приложения>:8009/emm/admin/ui
Консоль администратора ПУТ	http(s)://<сервер приложения>:8009/mt/admin/ui
Консоль администратора ПСУ	http(s)://<сервер приложения>:8009/push/admin/ui

3.8. Описание настройки подсистем ППО

3.8.1. Описание настройки ПСУ

Настройка ПСУ заключается в настройке обратного прокси-сервера (`reverse proxy`), а также в настройке протокола взаимодействия устройств (push-демона) с ПСУ.

3.8.1.1. Настройка обратного прокси-сервера

Обратный прокси-сервер (`reverse proxy`) служит для обработки запросов (подключений) мобильных устройств (push-демона) по защищенному протоколу TLS и транслирование этих запросов в ПСУ.

Примеры конфигурационных файлов обратного прокси-сервера Nginx для однонодовой и многонодовой конфигураций приведены в каталоге `samples/ac-mt/nginx_external-balancer/conf_stream.d/one-node.conf`.

Для настройки обратного прокси-сервера Nginx необходимо выполнить следующие действия:

3.8.1.1.1 Скопировать файл `samples/ac/nginx_external-balancer/conf_stream.d/one-node.conf` (или `samples/ac/nginx_external-balancer/conf_stream.d/three-node.conf` для многонодовой конфигурации) в каталог `/etc/nginx/conf_stream.d/` и переименовать его в `ocs-push-stream.conf` с помощью команды:

– для однонодовой конфигурации:

```
sudo cp samples/ac/nginx_external-balancer/conf_stream.d/one-node.conf
/etc/nginx/conf_stream.d/ocs-push-stream.conf
```

– для многонодовой конфигурации:

```
sudo cp samples/ac/nginx_external-balancer/conf_stream.d/three-
node.conf /etc/nginx/conf_stream.d/ocs-push-stream.conf
```

3.8.1.1.2 В секции `upstream` конфигурационного файла `ocs-push-stream.conf` задать адреса нод Серверов приложений ПСУ, например:

```
upstream internal-lb-stream-8025 {
  server ocs-app.local:8025 max_fails=3 fail_timeout=60 weight=1;
  least_conn;
}
```

3.8.1.1.3 Создать каталог для хранения лог-файлов Nginx:

```
mkdir -p <путь к каталогу>
```

Например:

```
mkdir -p /var/log/nginx/external_balancer/
```

3.8.1.1.4 В секции `server` задать значения следующих параметров:

– порт, к которому будут подключаться устройства, например:

```
listen 999 ssl so_keepalive=on;
```

– путь к закрытому ключу и сертификату закрытого ключа, которые будут использоваться для установки TLS-соединения, например:

```
ssl_certificate /etc/nginx/ssl/cert.pem;
ssl_certificate_key /etc/nginx/ssl/privkey.pem;
```

ПРИМЕЧАНИЕ. Сертификат закрытого ключа должен соответствовать следующим требованиям:

– в поле `Common Name (CN)` должно быть указано доменное имя;

– в сертификате должно присутствовать поле `subjectAltName`, значение которого должно совпадать со значением в поле `Common Name (CN)`;

– сертификат должен входить в цепочку доверия сертификатов на устройствах;

– файл сертификата закрытого ключа должен включать цепочку сертификатов.

Пример создания закрытого ключа и сертификата закрытого ключа приведен в пп. 3.8.1.2;

– путь к лог-файлам Nginx в соответствии с пп.3.8.1.1.3, например:

```
access_log /var/log/nginx/external_balancer/access-999.log basic;
error_log /var/log/nginx/external_balancer/error-999.log;
```

3.8.1.1.5 Проверить корректность конфигурационных файлов Nginx с помощью команды:

```
sudo nginx -t
```

В случае отсутствия ошибок будет выведено сообщение:

```
nginx: the configuration file /etc/nginx/nginx.conf syntax is ok
nginx: configuration file /etc/nginx/nginx.conf test is successful
```

3.8.1.1.6 Перезапустить Nginx с помощью команды:

```
sudo systemctl reload nginx
```

3.8.1.2. Пример создания закрытого ключа и сертификата закрытого ключа

ВНИМАНИЕ! Приведенный в настоящем пункте пример приведен исключительно в целях ознакомления с функционалом ПСУ. Создание и управление ключевой информацией должно осуществляться в соответствии с требованиями и регламентами эксплуатирующей организации.

Для формирования закрытого ключа и сертификата закрытого ключа, используемых для установки защищенного соединения обратного прокси-сервера с мобильными устройствами (push-демоном), необходимо выполнить следующие действия:

3.8.1.2.1 Создать закрытый ключ корневого сертификата с помощью команды:

```
openssl genpkey -algorithm RSA -out <путь к файлу с закрытым ключом> -
aes-128-cbc -pkeyopt rsa_keygen_bits:4096 -pass pass:
```

Например:

```
openssl genpkey -algorithm RSA -out rootCA.key -aes-128-cbc -pkeyopt
rsa_keygen_bits:4096 -pass pass:
```

3.8.1.2.2 Сформировать и подписать корневой сертификат с помощью команды:

```
openssl req -x509 -new -nodes -key <путь к файлу с закрытым ключом
корневого сертификата> -sha256 -days 1024 -out <путь к файлу с
корневым сертификатом>
```

Например:

```
openssl req -x509 -new -nodes -key rootCA.key -sha256 -days 1024 -out
rootCA.crt
```

3.8.1.2.3 Создать закрытый ключ сертификата обратного прокси-сервера с помощью команды:

```
openssl genpkey -algorithm RSA -out <путь к файлу с закрытым ключом
обратного прокси-сервера> -pkeyopt rsa_keygen_bits:4096
```

Например:

```
openssl genpkey -algorithm RSA -out server.key -pkeyopt
rsa_keygen_bits:4096
```

3.8.1.2.4 Создать конфигурационный файл, который будет использоваться для формирования запроса на сертификат прокси-сервера.

Конфигурационный файл должен соответствовать следующим требованиям:

– в параметре `Common Name (CN)` должно быть указано доменное имя;

– должен присутствовать параметр `subjectAltName`, значение которого должно совпадать со значением в поле `Common Name (CN)`.

Пример конфигурационного файла:

```
[ req ]
default_bits = 2048
prompt = no
default_md = sha256
req_extensions = req_ext
distinguished_name = dn

[ dn ]
C = RU
ST = Russia
L = Moscow
O = OMP
OU = OMP Aurora Center
CN = example.test

[ req_ext ]
subjectAltName = @alt_names

[ alt_names ]
DNS.1 = example.test

[ v3_ext ]
authorityKeyIdentifier=keyid,issuer:always
basicConstraints=CA:FALSE
keyUsage=keyEncipherment,dataEncipherment,digitalSignature
extendedKeyUsage=serverAuth,clientAuth
subjectAltName=@alt_names
```

Подробная информация о параметрах конфигурационного файла и порядке его формирования приведена в документации на криптографическую библиотеку OpenSSL: https://docs.openssl.org/3.0/man1/openssl-req/#configuration-file-format.

3.8.1.2.5 Создать запрос на сертификат обратного прокси-сервера с помощью команды:

```
openssl req -newkey rsa:4096 -keyout <путь к файлу с закрытым ключом обратного прокси-сервера> -out <путь к файлу с запросом на сертификат> -config <путь к конфигурационному файлу>
```

Например:

```
openssl req -newkey rsa:4096 -keyout server.key -out server.csr -
config csr.conf
```

3.8.1.2.6 Создать сертификат обратного прокси-сервера с помощью команды:

```
openssl x509 -req -in <путь к файлу с запросом на сертификат> -CA
<путь к файлу с корневым сертификатом> -CAkey <путь к файлу с закрытым
ключом корневого сертификата> -CAcreateserial -out <путь к файлу с
сертификатом обратного прокси-сервера> -days 500 -sha256 -extensions
v3_ext -extfile <конфигурационный файл>
```

Например:

```
openssl x509 -req -in server.csr -CA rootCA.crt -CAkey rootCA.key -
CAcreateserial -out server.crt -days 500 -sha256 -extensions v3_ext -
extfile csr.conf
```

3.8.1.2.7 Создать файл с цепочкой сертификатов. Для этого необходимо корневой сертификат записать в файл с сертификатом обратного прокси-сервера с помощью команды:

```
cat <путь к сертификату сервера>  <путь корневому сертификату> > <путь
к файлу цепочки.pem>
```

Например:

```
cat server.crt rootCA.crt > server.pem
```

3.8.1.2.8 Удалить пароль с закрытого ключа сертификата обратного прокси-сервера с помощью команды:

```
openssl rsa -in <путь к файлу с закрытым ключом обратного прокси-
сервера> -out <путь к файлу с закрытым ключом обратного прокси-сервера
без пароля>
```

Например:

```
openssl rsa -in server.key -out new.server.key
```

3.8.1.3. Настройка протокола взаимодействия устройств (push-демона) с ПСУ

Устройства в зависимости от установленной на них версии ОС Аврора используют разные версии протокола взаимодействия с ПСУ. Перечень версий протокола взаимодействия устройств с ПСУ приведен в таблице (Таблица 16).

Таблица 16

Версия протокола	Описание
0	Версия протокола для ОС Аврора 3-го поколения. Не поддерживает аутентификацию устройств в ПСУ
2	Версия протокола для ОС Аврора 4-го поколения, в которой учтены особенности многопользовательского режима ОС. Не поддерживает аутентификацию устройств в ПСУ
3	В данной версии протокола реализована аутентификация устройств при подключении к ПСУ. Поддерживается ОС Аврора версии 5.1 и выше. **ПРИМЕЧАНИЕ.** Использование протокола версии 3 допустимо при условии установки ПСУ совместно с ПУ и активации устройств в ПУ. В ином случае необходимо использовать протокол версии 2

Задание протокола осуществляется в параметре `allowedDeviceProtocolVersions` конфигурационного файла `install-<версия ППО>/install-ac-mt/config/subsystems/push/config.yml`. В данном параметре задается список доступных версий протокола. Если задано несколько версий, то будет использоваться максимальная версия протокола, которая поддерживается и устройством, и сервером. Примеры задания значений параметра `allowedDeviceProtocolVersions` приведены в таблице (Таблица 17).

Таблица 17

Значение параметра	Описание
`allowedDeviceProtocolVersions: []`	В ПСУ включена поддержка всех версий протокола. ПСУ аутентифицирует устройства, поддерживающие протокол версии 3. Устройства, не поддерживающие протокол версии 3 взаимодействуют с ПСУ без аутентификации. Данное значение используется по умолчанию

Значение параметра	Описание
`allowedDeviceProtocolVersions: [0, 2]`	Устройства взаимодействуют с ПСУ по протоколу версии 0 или версии 2. Аутентификация устройств не осуществляется
`allowedDeviceProtocolVersions: [3]`	Взаимодействовать с ПСУ могут только устройства, поддерживающие протокол версии 3. ПСУ осуществляет аутентификацию устройств

После внесения изменений необходимо переустановить конфигурационные файлы с помощью команды:

```
ANSIBLE_USER="<имя пользователя>" ./deploy-ac.sh --action config
```

3.8.2. Описание настройки ПУ

Настройка ПУ заключается в настройке взаимодействия Сервера приложений ПУ с Сервисом уведомлений Аврора версии 1.1.2 или ПСУ, а также в загрузке картографической информации в файловое хранилище ПУ.

3.8.2.1. Настройка взаимодействия Сервера приложений ПУ с ПСУ

В случае необходимости взаимодействия Сервера приложений ПУ с ПСУ потребуется выполнить следующие настройки:

3.8.2.1.1 Синхронизировать время между Сервером приложений ПУ и ПСУ.

3.8.2.1.2 Зарегистрировать в ПСУ проект и получить конфигурационные файлы с настройками: `mobile_app_ac_push_project.json` и `app_server_ac_push_project.json`. Инструкция по созданию проекта представлена в документе «Руководство пользователя. Часть 5. Подсистема Сервис уведомлений».

3.8.2.1.3 Задать протокол, домен и порт для обращения к ПСУ (значение параметра должно соответствовать значению параметра `push_public_address` в конфигурационном файле `app_server_ac_push_project.json`). Для этого в конфигурационном файле `config/config.yml.j2` задать параметр `config.publicUris.push.publicAddress`, например:

```
publicUris:
    push:
      publicAddress: "https://acenter.example.ru"
```

3.8.2.1.4 Задать домен <AC_DOMAIN> и порт ПСУ для устройств (push-демона).

Домен и порт ПСУ для устройств задаются в секции pushNotificationSystem конфигурационного файла config/config.yml.j2 и распространяются на все тенанты. По умолчанию в качестве mobileHostname указано имя первого сервера приложений.

```
   pushNotificationSystem:
    mobileHostname: "acenter.example.ru"
    mobilePort: 999
```

Также домен и порт можно задать в Консоли администратора ПУ при выполнении пп. 3.8.2.1.6. В данном случае настройки будут распространяться только на тенант в рамках которого выполнялась настройка.

ПРИМЕЧАНИЕ. Имя домена, задаваемое в параметре mobileHostname, должно совпадать с именем сервера адрес публичного API сервера авторизации, задаваемого в переменной publicUris.auth. publicAddress.

3.8.2.1.5 Переустановить ПУ в соответствии с п. 3.6.2, в случае если настройка осуществляется после установки ПУ.

3.8.2.1.6 В Консоли администратора ПУ («Администрирование» – «Настройки» – «Интеграция» – «Сервис уведомлений Аврора») задать параметры взаимодействия Сервера приложений ПУ и ПСУ.

Описания назначения параметров и порядок настройки приведены в документе «Руководство пользователя. Часть 3. Подсистема Платформа управления».

3.8.2.2. Ручная настройка доступа к приложению «Аврора Центр» для ОС Аврора и ОС Android

По умолчанию доступ к приложению «Аврора Центр» настраивается автоматически в процессе установки ППО.

Для ручной настройки доступа к приложению «Аврора Центр» необходимо выполнить следующие действия:

3.8.2.2.1 Выложить установочный файл (APK-файл или RPM-пакет) приложения «Аврора Центр» в файловое хранилище ПУ согласно параметру `root` секции `location /clientDownload` конфигурационного файла `/etc/nginx/conf.d/ocs-emm-static-files.nginx.conf` (по умолчанию каталог: `/ocs/emm/clients`), либо параметру `root` конфигурационного файла `install-<версия ППО>/install-ac-mt/config/subsystems/emm/applications/ocs-emm-static-files/ocs-emm-static-files.nginx.conf.j2` сценариев установки ППО.

Установочные файлы должны быть размещены следующим образом:

```
./clients/
├── android-<версия ОС>
│   └── client.noarch.apk
└── aurora-<версия ОС>
    └── client.<архитектура>.rpm
```

Например:

```
/ocs/emm/clients/
├── android-10
│   └── client.noarch.apk
└── aurora-5.0.0
    ├── client.arm.rpm
    └── client.arm64.rpm
```

В случае если имя установочного файла отличается от шаблона, то вместо него можно использовать символическую ссылку (`symbolic link`) на рядом лежащий файл. Для создания символической ссылки необходимо использовать команду:

```
ln -sf <относительный путь к apk-файлу> client.noarch.apk
или
ln -sf <относительный путь к rpm-пакету> client.rpm
```

Например:

```
ln -sf omp-emm-client-4.0.0.4+2-android.armeabi-v7a.apk
client.noarch.apk
```

Описание настройки файлового хранилища ПУ для размещения в нем установочных файлов приложения «Аврора Центр» приведено в п. 3.9.1.

3.8.2.2.2 В секции `config.provisioning.android.signatureChecksum` конфигурационного файла ПУ `install-<версия ППО>/install-ac-mt/config/subsystems/emm/config.yml` задать отпечаток сертификата ключа проверки ЭП в кодировке `base64`, с помощью которого будет выполняться проверка ЭП APK-файлов.

Например:

```
  provisioning:
    android:
      signatureChecksum: "lcspi4PJBHFyMPJATdZSNeQTy9L0kAPg2jwbO9L-Rw "
```

Получить отпечаток сертификата можно из APK-файла с помощью команды:

```
keytool -printcert -jarfile <имя файла> | perl -nle "print $& if
m{(?<=SHA256:) .*}" | xxd -r -p | openssl base64 | tr -d '=' | tr --
'+/=' '-_'
```

ПРИМЕЧАНИЕ. Утилита Keytool входит в состав Java SDK (или JRE).

3.8.2.2.3 Переустановить конфигурационные файлы сервиса `ocs-emm-enrollments-api` с помощью команды:

```
ANSIBLE_USER="<имя пользователя>" ./deploy-ac.sh --subsystems emm --
apps ocs-emm-enrollments-api --action config
```

3.8.2.3. Загрузка картографической информации в файловое хранилище ПУ

ПРИМЕЧАНИЕ. По умолчанию ППО настроено на работу с картой Москвы и Московской области. Для получения карты Евразии или новой версии карты необходимо обратится в службу технической поддержки предприятия-изготовителя.

Для загрузки картографической информации в хранилище ПУ, настроенное в соответствии с подразделом 3.9, необходимо выполнить следующие действия:

3.8.2.3.1 В файловое хранилище ПУ скопировать файл с картографической информацией и в соответствии с параметром `config.mbTilesSource` конфигурационного файла `config/subsystems/emm/applications/ocs-emm-locations-api/ocs-emm-locations-api.yml` (по умолчанию: `/ocs/emm/maps/map.osm.mbtiles`) создать символическую ссылку (`symbolic link`) на файл с картографической информацией с помощью следующих команд:

```
cp <имя файла> /ocs/emm/maps/
sudo ln -sf /ocs/emm/maps/<имя файла> /ocs/emm/maps/map.osm.mbtiles
```

Также допускает не использовать символическую ссылку, а размещать непосредственно сам файл в соответствии с параметром `config.mbTilesSource`.

3.8.2.3.2 Перезапустить сервис `ocs-emm-locations-api` с помощью команды:

```
ANSIBLE_USER="<имя пользователя>" ./deploy-ac.sh --subsystems emm --
apps ocs-emm-locations-api --action restart
```

3.8.3. Описание настройки ПООС

Настройка ПООС заключается в загрузке пакетов ОС в файловое хранилище.

Для загрузки пакетов ОС в файловое хранилище ПООС, настроенное в соответствии с подразделом 3.9, необходимо выполнить следующие действия:

3.8.3.1. Скопировать в произвольный каталог файлового хранилища ПООС архив с пакетами ОС и распаковать его в каталог, заданный в параметре `root` секции `location /pkgrepo/mobile` конфигурационного файла `/etc/nginx/conf.d/locations-external/ocs-pkgrepo-nginx-static.location` (по умолчанию каталог: `/ocs/pkgrepo/repos`), либо в параметре `repos_root` конфигурационного файла `install-<версия ППО>/install-ac-mt/config/subsystems/pkgrepo/vars/ocs-pkgrepo-nginx-static.yml` сценариев установки ППО:

```
tar -xf <имя файла с архивом> -C <путь к каталогу>
rm <имя файла с архивом>
```

Например,

```
tar -xf 4.0.2.35.tar -C /ocs/pkgrepo/repos
rm 4.0.2.35.tar
```

3.8.3.2. Зарегистрировать переданный релиз (версию), добавив в файл `/ocs/pkgrepo/meta/main.json` описание из специализированного meta-файла.

Meta-файл передается вместе с архивом репозитория обновления ОС и представляет собой файл в формате `.json` и имеет название `main.json`. Путь к meta-файлу задается в одном из следующих параметров:

– `alias` секции `location /pkgrepo/mobile/meta` конфигурационного файла `/etc/nginx/conf.d/locations-external/ocs-pkgrepo-nginx-static.location` (по умолчанию каталог: `/ocs/pkgrepo/meta`);

– `meta_root` конфигурационного файла `install-<версия ППО>/install-ac-mt/config/subsystems/pkgrepo/vars/ocs-pkgrepo-nginx-static.yml` сценариев установки ППО.

ВНИМАНИЕ! Приведенные в настоящем пункте примеры заполнения meta-файла приведены исключительно для общего ознакомления с возможной структурой файла. Итоговый meta-файл должен быть сформирован с учетом рекомендаций и примера заполнения, приведенных ниже.

Общие рекомендации по заполнению meta-файла:

– необходимо соблюдать общие правила структуры и синтаксиса формата json при создании meta-файла;

– необходимо корректно указывать следующие данные: модель устройства и версии ОС Аврора, до которых доступно обновление;

– следует придерживаться приведенных рекомендаций по заполнению файла;

– следует использовать инструменты для проверки синтаксиса подготовленного файла.

Meta-файл состоит из нескольких блоков, примеры заполнения которых приведены далее:

1) Общий блок:

```
{
  "brand": "OMP",
  "releases": []
}
```

где:

– "brand": "OMP" – общая информация;

– "releases": [] – блок по моделям;

2) Блок по моделям устройств:

```
{
      "deviceModel": "aq_ns220r",
      "latest": "4.0.2.249",
      "versions": [
        {
          "version": "4.0.2.249",
          "from": [
            "4.0.2.209"
          ]
        },
        {
          "version": "4.0.2.209",
          "from": [
            "4.0.2.175",
            "4.0.2.89"
          ]
        },
        {
          "version": "4.0.2.175",
          "from": [
            "4.0.2.89",
            "4.0.1.43",
            "4.0.1.20"
          ]
        },
        {
          "version": "4.0.2.89",
          "from": [
            "4.0.1.43",
            "4.0.1.20"
          ]
        },
        {
          "version": "4.0.1.43",
          "from": [
            "4.0.1.20"
          ]
        }
      ]
    }
```

где:

– "deviceModel": "aq_ns220r" – модель устройства Aquarius NS220 v5.2, представленная в кодовом наименовании: "aq_ns220r".

ПРИМЕЧАНИЕ. В случае если кодовое наименование устройства неизвестно следует запросить информацию у производителя:

– "latest": "4.0.2.249" – последняя доступная версия ОС Аврора для устройства;

– "versions": [] – блок списка версий;

3) Блок списка версий:

```
{
          "version": "4.0.2.249",
          "from": [
            "4.0.2.209"
          ]
        }
```

где:

– "version": "4.0.2.249" – необходимая версия ОС Аврора;

– "from": ["4.0.2.209"] – список версий ОС, с которых можно обновить устройство до необходимой версии ОС Аврора.

Пример заполненного meta-файла, составленный для устройства Aquarius NS220R с указанием возможности обновления ОС Аврора с версии 4.0.2.209 до версии 4.0.2.249:

```
{
   "brand":"OMP",
   "releases":[
      {
         "deviceModel":"aq_ns220r",
         "latest":"4.0.2.249",
         "versions":[
            {
               "version":"4.0.2.249",
               "from":[
                  "4.0.2.209"
               ]
            }
         ]
      }
   ]
}
```

3.8.3.3. Перезапустить сервис `ocs-pkgrepo-pkg-repo-api` с помощью команды:

```
ANSIBLE_USER="<имя пользователя>" ./deploy-ac.sh --subsystems pkgrepo
--apps ocs-pkgrepo-pkg-repo-api --action restart
```

3.8.3.4. Проверить корректность настройки, для чего необходимо войти в Консоль администратора ПУ, далее перейти в подраздел «Настройки» раздела «Администрирование», в раскрывающемся поле «Интеграция» выбрать вкладку «Обновление ОС» и убедиться, что отображаются имя сервера, модели устройств и доступные версии ОС (Рисунок 17).



Рисунок 17

3.8.4. Описание настройки CDN

Настройка CDN заключается в настройке контент-серверов.

Для настройки контент-серверов необходимо выполнить следующие действия:

3.8.4.1. Подготовить адрес, с которого будет раздаваться контент:

– завести отдельный домен для контент-сервера.

ПРИМЕЧАНИЕ. Количество адресов зависит от количества площадок, на которых будут размещены контент-серверы, а также от вариантов маршрутизации. Если за маршрутизацию будет отвечать гео-DNS/GSLB, то достаточно подготовить один адрес. Если маршрутизация будет осуществляться средствами ППО на основе IP-адресов клиентов, то необходимо подготовить отдельный адрес для каждой площадки;

– выпустить сертификат на заведенное доменное имя.

3.8.4.2. В секции `content` инвентарного файла `inventories/hosts.yml` задать адреса серверов (имена хостов), на которые будут установлены контент-серверы.

Например:

```
...
        content:
          hosts:
            acentercdn01:
            acentercdn02:
            acentercdn03:
```

Описание порядка задания адресов в инвентарном файле `inventories/hosts.yml` приведено в п. 3.10.13.

3.8.4.3. Раскомментировать секцию `content_servers_map` конфигурационного файла `config/vars/_vars.yml`.

3.8.4.4. В данной секции задать http адрес контент-сервера по умолчанию, параметр `content_servers_map.default`:

```
...
content_servers_map:
  default: "<адрес контент сервера>"
```

На контент-сервер по умолчанию будут перенаправлять запросы из сетей/подсетей отсутствующих в секции `content_servers_map.content_servers`.

3.8.4.5. В секции `content_servers_map.content_servers` конфигурационного файла `config/vars/_vars.yml`, при необходимости, задать правила перенаправления запросов на контент-серверы (т.е. задать из каких сетей/подсетей на какие контент-серверы будут перенаправляться запросы):

```
...
  content_servers:
    - server: "<адрес контент сервера>"
      addresses:
        - "<адрес сети/подсети>"
        - "<адрес сети/подсети>"
```

Например:

```
...
  content_servers:
    - server: "https://ocs-cdn01.test.ru"
      addresses:
        - "192.168.79.128"
        - "192.168.79.133"
    - server: "https://ocs-cdn02.test.ru"
      addresses:
        - "192.168.0.0/16"
```

3.8.4.6. В секции `content_servers_map.delete` конфигурационного файла `config/vars/_vars.yml` при необходимости задать адреса сетей/подсетей, на которые не должны распространяться правила из секции `content_servers_map.content_servers`.

3.8.4.7. В секции `publicUris.cdn.originAddress` конфигурационного файла `config/config.yml.j2` указать адрес сервер оригинации для контент-сервера (адрес сервера приложения ППО).

Например:

```
...
  publicUris:
    cdn:
      originAddress: "https://aurora-center.test.ru:8443"
```

ВНИМАНИЕ! Сетевой доступ к указанному адресу должен быть открыт с контент-серверов (см. схемы развертывания п. 2.3.5, п. 2.3.6).

Установка компонентов контент-сервера выполняется вместе с установкой ППО (подраздел 3.6) при условии, что предварительно были выполнены описанные выше настройки контент-сервера.

ПРИМЕЧАНИЕ. Для обеспечения отказоустойчивости контент-сервер может быть развернут в многонодовой конфигурации с внешним балансировщиком нагрузки. Для этого рекомендуется воспользоваться информацией, приведенной в п. 3.10.10, а также примером конфигурационного файла внешнего балансировщика, приведенного в `samples/ac-mt/nginx_external-balancer/conf_stream.d/content-server.conf`.

3.9. Описание настройки файлового хранилища ППО

3.9.1. Настройка файловых хранилищ подсистем ППО

Для настройки файловых хранилищ подсистем ППО необходимо выполнить следующие действия:

3.9.1.1. Создать каталог `/ocs` и назначить его владельцем пользователя `ocs`, под которым работают сервисы ППО:

```
sudo mkdir -p /ocs
sudo chown ocs:ocs /ocs
```

3.9.1.2. В случае использования единого файлового хранилища необходимо выполнить монтирование данного хранилища к каталогу `/ocs`.

ВНИМАНИЕ! При эксплуатации ППО в кластерной конфигурации все ноды Сервера приложений ППО с ПМ, ПУ и ПООС должны иметь доступ к файловому хранилищу. Соответственно, все ноды Сервера приложений ППО должны быть настроены на работу с данным файловым хранилищем.

Варианты и порядок настройки доступа нод Сервера приложений ППО к файловому хранилищу приведены в п. 3.9.2.

3.9.1.3. Настроить файловое хранилище ПМ, в котором будут храниться файлы приложений (иконки, скриншоты, RPM-пакеты), загружаемые разработчиками.

Для этого в каталоге `/ocs` необходимо создать каталог в соответствии с параметром `filestoragePath` конфигурационного файла `config/subsystems/appstore/config.yml`. В созданном каталоге потребуется создать каталог `applications-api` и назначить его владельцем пользователя `ocs`, под которым работают сервисы ПМ:

```
sudo mkdir -p /ocs/appstore/applications-api
sudo chown ocs:ocs /ocs/appstore/applications-api
```

Параметр `filestoragePath` конфигурационного файла `config/subsystems/appstore/config.yml` может иметь следующий вид:

```
filestoragePath: "/ocs/appstore"
```

Для того, чтобы пакеты обновлений ОС сохранялись между перезапусками контейнера, необходимо создавать директорию на хост-машине (например, `mkdir -p pkgrepo`) и смонтировать ее в контейнер при запуске. Добавить в команду запуска `docker run` следующий параметр:

```
--volume $(pwd)/pkgrepo:/ocs/pkgrepo/repos
```

ПРИМЕЧАНИЕ. `/ocs/pkgrepo/repos` – внутренний путь к хранилищу ПООС.

3.9.1.4. Настроить файловое хранилище ПУ, в котором будут храниться установочные файлы приложения «Аврора Центр» и картографическая информация.

Для этого необходимо выполнить следующие действия:

3.9.1.4.1 В каталоге `/ocs` необходимо в соответствии с параметром `root` секции `location /clientDownload` конфигурационного файла `/etc/nginx/conf.d/ocs-emm-static-files.nginx.conf` (по умолчанию каталог: `/ocs/emm/clients`), либо параметром root конфигурационного файла `install-<версия ППО>/install-ac-mt/config/subsystems/emm/applications/ocs-emm-static-files/ocs-emm-static-files.nginx.conf.j2` сценариев установки ППО

создать каталог и назначить его владельцем пользователя ocs, под которым работают сервисы ПУ:

```
sudo mkdir -p <путь к каталогу>
sudo chown ocs:ocs <путь к каталогу>
```

Например:

```
sudo mkdir -p /ocs/emm/clients
sudo chown ocs:ocs /ocs/emm/clients
```

3.9.1.4.2 В каталоге /ocs необходимо в соответствии с параметром config.mbTilesSource конфигурационного файла config/subsystems/emm/applications/ocs-emm-locations-api/ocs-emm-locations-api.yml (по умолчанию: /ocs/emm/maps/map.osm.mbtiles) создать каталог и назначить его владельцем пользователя ocs, под которым работают сервисы ПУ:

```
sudo mkdir -p <путь к каталогу>
sudo chown ocs:ocs <путь к каталогу>
```

Например:

```
sudo mkdir -p /ocs/emm/maps
sudo chown ocs:ocs /ocs/emm/maps
```

3.9.1.5. Настроить файловое хранилище ПООС, в котором будут храниться пакеты ОС.

Для этого в каталоге /ocs необходимо создать каталог согласно параметру root секции location /pkgrepo/mobile конфигурационного файла /etc/nginx/conf.d/locations-external/ocs-pkgrepo-nginx-static.location (по умолчанию каталог: /ocs/pkgrepo/repos), либо параметру repos_root конфигурационного файла install-<версия ППО>/install-ac-mt/config/subsystems/pkgrepo/vars/ocs-pkgrepo-nginx-static.yml сценариев установки ППО:

```
mkdir -p <путь к каталогу>
```

Например,

```
mkdir -p /ocs/pkgrepo/repos
```

3.9.2. Настройка доступа нод сервера приложений ППО к файловому хранилищу

При эксплуатации ППО в кластерной конфигурации все ноды сервера приложений ППО должны иметь доступ к файловому хранилищу, в котором располагаются файлы приложений, пакеты ОС, картографическая информация. Доступ нод сервера приложений к файловому хранилищу может быть организован следующими способами:

– синхронизация файлов между файловыми хранилищами каждой ноды сервера приложений с помощью приложения `Syncthing`;

– использование единого файлового хранилища.

3.9.2.1. Настройка синхронизации файлов между файловыми хранилищами каждой ноды сервера приложений с помощью приложения Syncthing

Приложение `Syncthing` выполняет синхронизацию файлов в режиме реального времени между нодами сервера приложений ППО.

Для настройки и установки `Syncthing` необходимо выполнить следующие действия:

3.9.2.1.1 Придумать пароль пользователя (по умолчанию пользователь `ocs`) для доступа к графическому интерфейсу приложения.

3.9.2.1.2 Сформировать хэш-код пароля с использованием алгоритма `bcrypt` с помощью следующих команд:

```
sudo apt install python3-passlib python3-bcrypt
ansible all -i localhost, -m debug -a "msg={{ '<пароль>' |
password_hash('bcrypt') }}"
```

3.9.2.1.3 В параметре `syncthing.gui_password` конфигурационного файла `config/secret.yml` задать значение хэш-кода пароля:

```
  syncthing:
    gui_password:
'$2a$10$N.9m94jj3ciTDtlUhxudwu2rGE3jgb4A0GUCT30KsUIEIbdcPZIx6'
```

3.9.2.1.4 В параметре `syncthing.gui_apikey` конфигурационного файла `config/secret.yml` ключ доступа к API приложения:

```
    gui_apikey: 'CHANGEME-AgpY4dv2tdcwcNXSmhnxW55euHD55Eyf'
```

3.9.2.1.5 В секции `syncthing_folders` конфигурационного файла `config/vars/_vars.yml` при необходимости изменить предустановленные значения каталогов, для которых требуется выполнять синхронизацию:

```
syncthing_folders:
  pkgrepo:
    path: /ocs/pkgrepo
  appstore:
    path: /ocs/appstore
  emm:
    path: /ocs/emm
```

3.9.2.1.6 Установить приложение с помощью команды:

```
ANSIBLE_USER="<имя пользователя>" ./deploy-infra.sh -c syncthing
```

Например:

```
ANSIBLE_USER="omp" ./deploy-infra.sh -c syncthing
```

ВНИМАНИЕ! При установке приложения осуществляется генерация ключевой пары, которая используется для защищенного обмена данными между экземплярами приложения, развернутыми на серверах. Для смены ключевой пары (например, в случае ее компрометации), необходимо удалить приложение и установить его заново.

Удаление приложения осуществляется с помощью команды:

```
ANSIBLE_USER="<имя пользователя>" ./deploy-infra.sh -c syncthing -A
flush_all
```

3.9.2.2. Пример настройки единого файлового хранилища

Единое файловое хранилище применяется для хранения файлов приложений (иконки, скриншоты, RPM-пакеты) и пакетов ОС.

Для настройки единого файлового хранилища необходимо выполнить следующие действия:

3.9.2.2.1 Установить NFS сервер в соответствии с официальной документацией на ОС RedHat, приведенной на странице: `https://access.redhat.com/documentation/en-us/red_hat_enterprise_linux/7/html/storage_administration_guide/nfs-serverconfig`.

3.9.2.2.2 На Серверах приложений ПМ и ПООС создать каталог `/ocs` и назначить его владельцем пользователя `ocs`, под которым работают сервисы ПМ и ПООС:

```
sudo mkdir -p /ocs
sudo chown ocs:ocs /ocs
```

ВНИМАНИЕ! При эксплуатации ППО в кластерной конфигурации все ноды серверов приложений ПМ и ПООС должны иметь доступ к единому файловому хранилищу. Соответственно, все ноды серверов приложений ПМ и ПООС должны быть настроены на работу с данным файловым хранилищем.

3.9.2.2.3 Выполнить монтирование файловой системы NFS к каталогу `/ocs` с помощью команды:

```
mount example.com:/export/ocsfs /ocs
```

где:

– `example.com` – имя узла файлового сервера NFS;

– `/export/ocsfs` – каталог, который экспортирует `example.com`;

– `/ocs` – каталог, к которому осуществляется монтирование.

ПРИМЕЧАНИЕ. Монтирование файловой системы NFS также может быть выполнено посредством редактирования файла `/etc/fstab`. Для этого в данный файл необходимо добавить запись следующего вида:

```
example.com:/export/ocsfs /ocs nfs defaults 0 0
```

Редактирование файла `/etc/fstab` должно осуществляться суперпользователем.

3.9.2.2.4 Для проверки корректности монтирования необходимо выполнить команду:

```
ls /ocs
```

и убедиться, что полученный список файлов соответствует списку файлов в каталоге `/export/ocsfs` на компьютере `example.com`.

3.10. Дополнительные настройки ППО и среды функционирования ППО

3.10.1. Настройка взаимодействия сервера приложений ПУ с SMTP-сервером

Настройка взаимодействия Сервера приложений ПУ с SMTP-сервером требуется для обеспечения отправки на электронную почту писем с QR-кодами для активации устройств.

Для настройки взаимодействия Сервера приложений ПУ с SMTP-сервером необходимо в секции `smtp` конфигурационного файла подсистемы ПУ `config.yml (config/subsystems/emm/config.yml)` задать требуемые значения:

– адрес электронной почты, с которого отправляются письма (параметр: `from`);

– адрес сервера электронной почты (параметр: `address`);

– флаг включения/отключения использования защищенного протокола SMTPS для защиты соединения (параметр: `tls`);

– тип аутентификации (параметр: `authType`);

– параметры для заданного типа аутентификации (`username`, `password` и др.). Значения параметров `from` и `username` должны быть идентичны, в противном случае почтовый сервер будет отклонять сообщения. С помощью флага «`tls`» настраивается необходимость использования протокола SMTPS для защиты соединения между Сервером приложений ПУ и SMTP-сервером. При значении флага

«tls: true» будет использоваться защищенный протокол SMTPS. Если SMTP-сервер не поддерживает протокол SMTPS, то Сервер приложений ПУ вернет ошибку «tls: first record does not look like a TLS handshake» и сообщение отправлено не будет. При значении флага «tls: false» будет использоваться протокол SMTP. В случае, если SMTP-сервер поддерживает расширение STARTTLS, то передача сообщений будет осуществляться с использованием протокола TLS. В ПУ поддерживаются LOGIN, CRAM-MD5, PLAIN типы аутентификации SMTP. В зависимости от используемого типа аутентификации необходимо задать следующие параметры (остальные параметры оставить без изменений):

- LOGIN:

```
smtp:
  from:"user@example.com"
  address:"smtp.example.com:1025"
  tls:true
  authType:"LOGIN"
  username:"user@example.com"
  password: "test_password"
```

- CRAM-MD5:

```
smtp:
  from:"user@example.com"
  address:"smtp.example.com:1025"
  tls:true
  authType:"CRAM-MD5"
  username:"user@example.com"
  secret: "test_secret"
```

- без аутентификации:

```
smtp:
  from:"user@example.com"
  address:"smtp.example.com:1025"
  tls: true
```

- PLAIN:

```
smtp:
  from: "user@example.com"
  address: "smtp.example.com:1025"
  tls: true
  authType: "PLAIN"
```

```
  host: "smtp.example.com"
  username: "user@example.com"
  password: "test_password"
  identity: "identity"
```

ПРИМЕЧАНИЕ. Для аутентификации типа PLAIN в поле «`host`» должно быть указано значение из поля «`address`» без указания порта. Также для данного типа аутентификации необходимо наличие защищенного TLS-соединения. Если TLS-соединение не поддерживается, то при попытке отправить сообщение возникнет ошибка `error: unencrypted connection` и сообщение не будет отправлено.

После изменения настроек необходимо переустановить конфигурационные файлы с помощью команды:

```
ANSIBLE_USER="<имя пользователя>" ./deploy-ac.sh --subsystems emm --
apps ocs-emm-dispatcher-api,ocs-emm-enrollments-api --action config
```

3.10.2. Настройка взаимодействия сервера приложений ПУ с сервером удаленного подключения RustDesk

Сервер RustDesk используется для удаленного подключения к рабочему столу активированного устройства, а также для обмена файлами с помощью ПУ.

ВНИМАНИЕ! Данный функционал доступен только для устройств, функционирующих под управлением ОС Android и ОС Альт Linux.

ПРИМЕЧАНИЯ:

– рекомендуется использовать сервер RustDesk версии 1.4.9;

– инструкция по установке расположена на официальном сайте: https://rustdesk.com/docs/en/self-host/rustdesk-server-oss/install/.

Схема взаимодействия ППО и RustDesk приведена на рисунке (Рисунок 18).

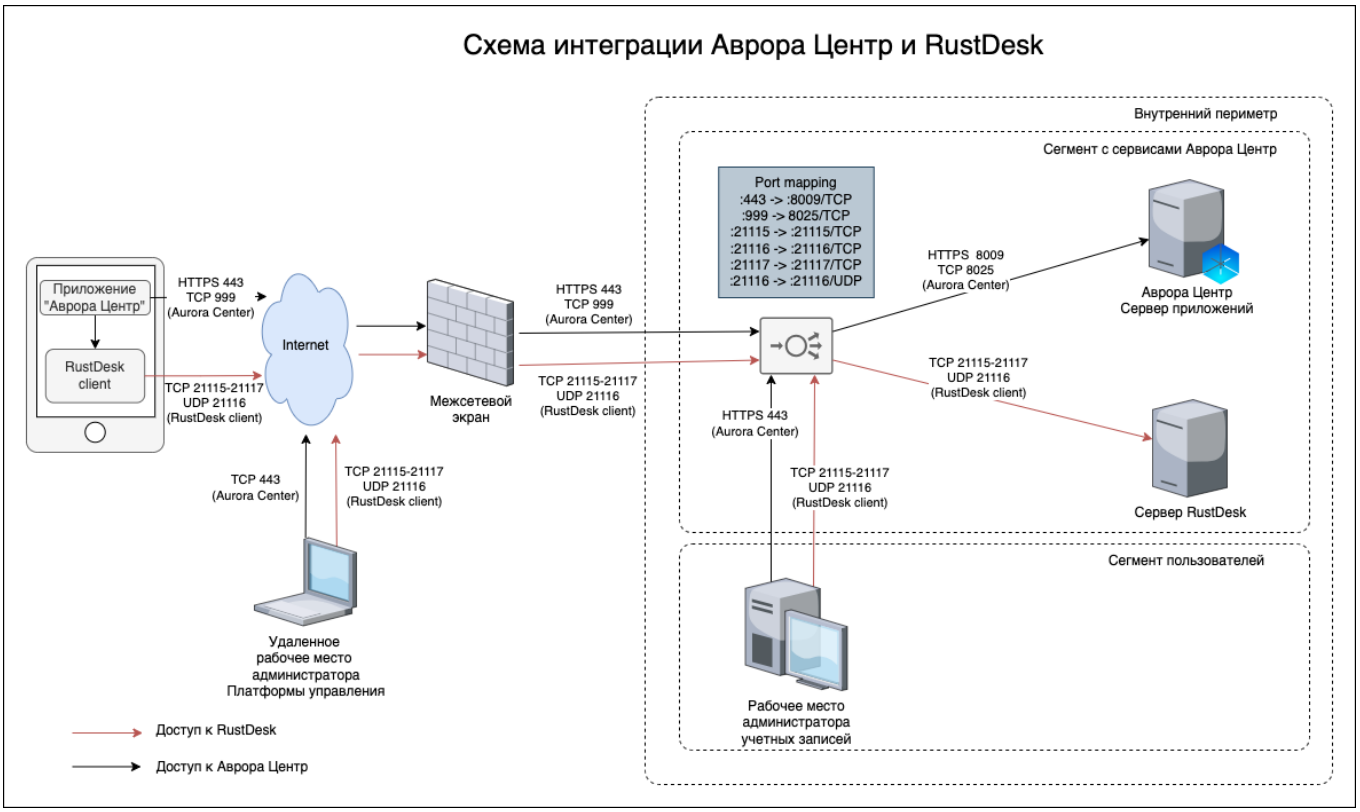


Рисунок 18

Для настройки взаимодействия Сервера приложений ПУ с сервером RustDesk необходимо в секции `remoteControl` конфигурационного файла `config/config.yml.j2` задать требуемые значения:

– адрес сервера RustDesk (параметр: `serverUrl`);

– ключ доступа к серверу RustDesk (параметр: `serverKey`).

Например:

```
  remoteControl:
    serverUrl: "rustdesk.server:21116"
    serverKey: "<RUSTDESK_KEY>"
    sessionTimeout: "30m"
```

После изменения настроек необходимо переустановить конфигурационные файлы с помощью команды:

```
ANSIBLE_USER="<имя пользователя>" ./deploy-ac.sh --subsystems emm --
action config
```

3.10.3. Настройка разделения трафика

ППО позволяет разделять входящий трафик (URL-запросы) следующими способами:

– по `basepath` – каждая Консоль администратора/разработчика (либо API для взаимодействия с приложениями) привязана к определенному `basepath`. `Basepath` заданы в секции `config.publicUris` конфигурационного файла `internal.yml`;

– по доменам (субдоменам) – каждая Консоль администратора/разработчика и API для взаимодействия с приложениями (либо группа консолей и API) опционально может быть привязана к определенному домену. Рекомендуется публичные консоли и API привязывать к домену, который имеет доступ из сети Интернет, а внутренние консоли (Консоли администраторов) привязывать к домену, не имеющему доступ из сети Интернет. Разделение трафика по доменам приведено в пп. 3.5.2.3;

– по портам – внутренние и внешние адреса ППО привязаны к отдельным портам. Описание настройки разделения трафика по портам приведено в пп. 3.5.2.2.

3.10.4. Настройка кэширования ответов сервисов

Для увеличения производительности ППО применяется кэширование ответов сервисов с помощью Nginx. При этом доступ к закэшированным данным осуществляется через шлюзы доступа ППО.

Настройки кэширования задаются в следующих конфигурационных файлах сценариев установки среды функционирования ППО и инфраструктурных компонентов ППО:

1) В конфигурационном файле `shared_roles/nginx/defaults/main.yml` задаются следующие параметры:

– `cache_enabled` – включение/выключение кэширования;

– `cache_path` – каталог хранения кэша;

– `cache_keys_zone` – имя зоны в разделяемой памяти, где будет храниться кэш;

– `cache_keys_zone_size` – размер зоны в разделяемой памяти;

– `cache_max_size` – максимальный размер выделяемой под кэш памяти (когда место заканчивается, Nginx удаляет устаревшие данные);

– `cache_inactive` – время, после которого кэш будет автоматически очищаться.

Например:

```
cache_enabled: true
cache_path: "/var/cache/nginx"
cache_keys_zone: "proxy_cache"
cache_keys_zone_size: "50m"
cache_max_size: "10G"
cache_inactive: "30m"
```

ПРИМЕЧАНИЕ. Максимальный размер выделяемой под кэш памяти должен быть не менее 10 ГБ;

2) В конфигурационных файлах `config/subsystems/<название подсистемы>/vars/services.yml` задаются API функции (endpoint-ы) ППО, для которых необходимо выполнять кэширование, а также параметры кэширования для каждой API функции:

– `proxy_cache` – включение кэширования для API функции;

– `proxy_cache_valid` – время кэширования ответа (возможно задать время кэширования для определенных статусов ответа);

– `proxy_cache_lock` – параметр определяет возможность прохождения нескольких запросов на бэкенд (к сервисам ППО). При значении «on» запрещается прохождение нескольких запросов к сервису ППО, все повторные запросы будут ожидать появления ответа в кэше либо таймаута блокировки запроса к странице;

– `proxy_cache_use_stale` – параметр определяет, в каких случаях можно использовать устаревший закэшированный ответ;

– `add_header: "X-Cache-Status $upstream_cache_status"` – директива добавляет http-заголовок, содержащий статус кэширования.

Например:

```
...
      nginx_location_dashboard:
        path: "~ /v1/dashboards/[^/]+$"
        proxy_cache: "proxy_cache"
        proxy_cache_valid: "200 {{ cache_interval_dynamic }}"
        proxy_cache_lock: "on"
        proxy_cache_use_stale: "updating"
        proxy_cache_background_update: "on"
        add_header: "X-Cache-Status $upstream_cache_status"
```

3.10.5. Действия по безопасной установке и настройке средства

ПРИМЕЧАНИЕ. Установка, настройка и эксплуатация ППО должна осуществляться в соответствии с ЭД на ППО.

При использовании ППО в государственных информационных системах (ГИС) (информационных системах персональных данных, автоматизированных системах управления, критической информационной инфраструктуре), не содержащих информации, составляющей государственной тайны, в зависимости от класса защищенности должны быть установлены значения параметров, приведенные в таблице (Таблица 18).

Таблица 18

Параметр	Значение (для ГИС 4-го класса)	Значение (для ГИС 3-го класса)	Значение (для ГИС 2-го класса)	Значение (для ГИС 1-го класса)
Конфигурационный файл ПБ (сценария установки ПБ): `/var/ocs/config/subsystems/auth/config.yml` (`config/subsystems/auth/config.yml`)				
Период времени неиспользования идентификатора (учетной записи) пользователя, через которое происходит его блокирование: `config.maxAccountInactivityPeriod`	Устанавливается на усмотрение оператора ИС, например: `maxAccountInactivityPeriod: 2160h`	Не более 90 дней, например: `maxAccountInactivityPeriod: 2160h`	Не более 90 дней, например: `maxAccountInactivityPeriod: 2160h`	Не более 45 дней, например: `maxAccountInactivityPeriod: 1080h`
Минимальная длина пароля: `config.passwordSettings.minLength`	Не менее 6 символов, например: `config.passwordSettings.minLength: 6`	Не менее 6 символов, например: `config.passwordSettings.minLength: 6`	Не менее 6 символов, например: `config.passwordSettings.minLength: 6`	Не менее 8 символов, например: `config.passwordSettings.minLength: 8`

Алфавит пароля для учетных записей пользователей не настраивается. Пароли учетных записей пользователей должны содержать буквы верхнего и нижнего регистров, цифры и	Не менее 30 символов, например:
	`minDigits: 1`
	`minUpperLetters: 0`
	`minLowerLetters: 1`
		Не менее 60 символов, например:
		`minDigits: 1`
		`minUpperLetters: 1`
		`minLowerLetters: 1`
			Не менее 70 символов, например:
			`minDigits: 1`
			`minUpperLetters: 1`
			`minLowerLetters: 1`
				Не менее 70 символов, например:
				`minDigits: 1`
				`minUpperLetters: 1`
				`minLowerLetters: 1`

Параметр	Значение (для ГИС 4-го класса)	Значение (для ГИС 3-го класса)	Значение (для ГИС 2-го класса)	Значение (для ГИС 1-го класса)

специальные символы (это контролируется ППО).
Алфавит пароля для учетных записей устройств:
– минимальное число цифр в пароле:
`config.passwordSettings.minDigits`
– минимальное число букв верхнего регистра в пароле:
`config.passwordSettings.minUpperLetters`
– минимальное число букв нижнего регистра в пароле:
`config.passwordSettings.minUpperLetters`
– минимальное число спецсимволов в пароле:
`config.passwordSettings.minSpecialChars`
	`minSpecialChars: 0`	`minSpecialChars: 0`	`minSpecialChars: 1`	`minSpecialChars: 1`

Параметр	Значение (для ГИС 4-го класса)	Значение (для ГИС 3-го класса)	Значение (для ГИС 2-го класса)	Значение (для ГИС 1-го класса)
Максимальное время действия пароля: `config.passwordExpirationTime`	Не более 180 дней, например: `passwordExpirationTime: "4320h"`	Не более 120 дней, например: `passwordExpirationTime: "2880h"`	Не более 90 дней, например: `passwordExpirationTime: "2160h"`	Не более 60 дней, например: `passwordExpirationTime: "1440h"`
Максимальное время действия ключа учетной записи сервера приложений: `config.ttl.client_jwks`	Не более 1 года и 3 мес., например: `client_jwks: 10950h`	Не более 1 года и 3 мес., например: `client_jwks: 10950h`	Не более 1 года и 3 мес., например: `client_jwks: 10950h`	Не более 1 года и 3 мес., например: `client_jwks: 10950h`
Число последних использованных паролей, которые запрещено использовать пользователями при создании новых паролей: `config.passwordHistoryDepth`	Устанавливается на усмотрение оператора ИС, например: `passwordHistoryDepth: 3`	Устанавливается на усмотрение оператора ИС, например: `passwordHistoryDepth: 3`	Устанавливается на усмотрение оператора ИС, например: `passwordHistoryDepth: 3`	Устанавливается на усмотрение оператора ИС, например: `passwordHistoryDepth: 3`
Максимальное количество неуспешных попыток аутентификации (ввода неправильного пароля) до блокировки: `config.failedLoginTries`	От 3 до 10 попыток, например: `failedLoginTries: 10`	От 3 до 10 попыток, например: `failedLoginTries: 10`	От 3 до 8 попыток, например: `failedLoginTries: 8`	От 3 до 4 попыток, например: `failedLoginTries: 4`

Параметр	Значение (для ГИС 4-го класса)	Значение (для ГИС 3-го класса)	Значение (для ГИС 2-го класса)	Значение (для ГИС 1-го класса)
Время блокировки учетной записи пользователя в случае достижения установленного максимального количества неуспешных попыток аутентификации: `config.failedLoginBlockTime`	От 3 до 15 минут, например: `failedLoginBlockTime: "3m"`	От 5 до 30 минут, например: `failedLoginBlockTime: "5m"`	От 10 до 30 минут, например: `failedLoginBlockTime: "10m"`	От 15 до 60 минут, например: `failedLoginBlockTime: "15m"`
Количество одновременных сессий для привилегированных учетных записей: `config.privilegedSessionsLimit`	Устанавливается на усмотрение оператора ИС, например: `privilegedSessionsLimit: 10`	Устанавливается на усмотрение оператора ИС, например: `privilegedSessionsLimit: 10`	Устанавливается на усмотрение оператора ИС, например: `privilegedSessionsLimit: 10`	Не более 2-х, например: `privilegedSessionsLimit: 2`
Количество одновременных сессий для непривилегированных учетных записей: `config.unprivilegedSessionsLimit`	Устанавливается на усмотрение оператора ИС, например: `unprivilegedSessionsLimit: 10`	Устанавливается на усмотрение оператора ИС, например: `unprivilegedSessionsLimit: 10`	Устанавливается на усмотрение оператора ИС, например: `unprivilegedSessionsLimit: 10`	Устанавливается на усмотрение оператора ИС, например: `unprivilegedSessionsLimit: 10`

Параметр	Значение (для ГИС 4-го класса)	Значение (для ГИС 3-го класса)	Значение (для ГИС 2-го класса)	Значение (для ГИС 1-го класса)
Общий конфигурационный файл ППО (шаблон общего конфигурационного файла ППО): `/var/ocs/config/config.yml (config/config.yml.j2)`				
Время бездействия (неактивности) пользователя, через которое осуществляется завершение сеанса пользователя: `config.session.rememberFor`	Устанавливается на усмотрение оператора ИС, например: `rememberFor: 30m`	Устанавливается на усмотрение оператора ИС, например: `rememberFor: 30m`	Не более 15 минут, например: `rememberFor: 15m`	Не более 5 минут, например: `rememberFor: 5m`

3.10.6. Действия по смене аутентификационной информации (паролей, секретов, токенов, ключей)

При эксплуатации ППО должна обеспечиваться периодическая смена аутентификационной информации. Периодичность смены определяется эксплуатирующей организацией. Смена аутентификационной информации также должна осуществляться в случае ее компрометации. К событиям компрометации относятся (но не ограничиваются), следующие события:

– НСД к серверам приложений ППО и/или управляющей ЭВМ;

– потеря носителя, содержащего аутентификационную информацию;

– увольнение сотрудников, имевших доступ к аутентификационной информации;

– возникновение подозрений на утечку аутентификационной информации;

– случаи, когда нельзя достоверно установить, что произошло с носителем аутентификационной информации (например, не понятна причина выхода носителя из строя).

Аутентификационная информация компонентов среды функционирования ППО, инфраструктурных компонентов ППО, а также секретный ключ клиентов (сервисов) и ключ шифрования секретов задаются в конфигурационных файлах `config/vars/_vars.yml` и `config/secret.yml`. Для смены, указанной аутентификационной информации необходимо выполнить следующие действия:

3.10.6.1. Изменить пароли, секреты, токены в конфигурационных файлах `config/vars/_vars.yml` и `config/secret.yml`.

3.10.6.2. Установить компоненты среды функционирования ППО и инфраструктурные компоненты ППО в соответствии с п. 3.6.1.

3.10.6.3. Установить ППО в соответствии с п. 3.6.2.

3.10.7. Действия по реализации функций безопасности среды функционирования ППО и инфраструктурных компонентов ППО

3.10.7.1. Требования к межсетевому экранированию

Необходимо, чтобы защита периметра (физических или логических границ) ИС осуществлялась с использованием межсетевого экрана требуемого класса защиты.

Межсетевой экран должен пропускать трафик только на внешние порты ППО, при этом остальной трафик должен быть запрещен. Перечень внешних портов ППО в зависимости от варианта настройки приведен в таблице (Таблица 19).

Таблица 19

Номер порта (протокол)	Описание	Конфигурационный файл, в котором задается порт	Тип порта[20]
Сервисы ППО «Аврора Центр»			
10000 – 10500 (tcp)	Порты сервисов ППО	`shared_roles/systemd-deploy/templates/systemd-supplicant.sh.j2 /usr/bin/systemd-supplicant.sh`	внутренний
Nginx			
80 (tcp)	Служит для взаимодействия сервисов ППО друг с другом	`shared_roles/consul-template/defaults/main.yml`	внутренний
999 (tls)	Служит для взаимодействия устройств с ПСУ	Конфигурационный файл Nginx согласно п. 3.8.1	внешний
8007	Служит для межподсистемного взаимодействия.	`config/vars/_vars.yml`	внутренний
8009 (tcp)	Балансировщик сервисов (Nginx Web Server)	`config/vars/_vars.yml config/config.yml.j2 /etc/nginx/conf.d/ocs.conf`	внешний
8024 (tcp)	Порт для приема запросов от контент-серверов	`config/vars/_vars.yml`	внешний

[20] Описание типов портов приведено в таблице (Таблица 36).

Номер порта (протокол)	Описание	Конфигурационный файл, в котором задается порт	Тип порта[20]
8025 (tcp)	На данный порт перенаправляются запросы с 999 порта	Конфигурационный файл Nginx согласно п. 3.8.1	внутренний
8081 (tcp)	Служит для сбора метрик на шлюзах доступа	`shared_roles/telemetry/defaults/main.yml`	внутренний
СУБД PostgreSQL			
5432 (tcp)	СУБД PostgreSQL	`shared_roles/postgresql/defaults/main.yml`	внутренний
СУБД Valkey			
6379 (tcp)	`valkey-server`	`shared_roles/valkey/defaults/main.yml`	внутренний
26379 (tcp)	`valkey-sentinel`	`shared_roles/valkey/defaults/main.yml`	внутренний
Consul			
8300 (tcp)	https://www.consul.io/docs/install/ports		внутренний
8301 (tcp/udp)	https://www.consul.io/docs/install/ports		внутренний
8302 (tcp/udp)	https://www.consul.io/docs/install/ports		внутренний
8600 (tcp/udp)	https://www.consul.io/docs/install/ports	`shared_roles/consul/defaults/main.yml`	внутренний
8500 (tcp)	https://www.consul.io/docs/install/ports	`shared_roles/consul/defaults/main.yml`	внутренний
Redpanda			
8080 (tcp)	`redpanda_console_port`	`shared_roles/redpanda/defaults/main.yml`	внутренний
9092 (tcp)	`redpanda_kafka_api`	`shared_roles/redpanda/defaults/main.yml`	внутренний
9644 (tcp)	`redpanda_admin`	`shared_roles/redpanda/defaults/main.yml`	внутренний
33145 (tcp)	`redpanda_rpc_server`	`shared_roles/redpanda/defaults/main.yml`	внутренний
Syncthing			
8384 (tcp)	https://docs.syncthing.net/users/firewall.html	`shared_roles/syncthing/defaults/main.yml`	внутренний

Номер порта (протокол)	Описание	Конфигурационный файл, в котором задается порт	Тип порта[20]
22000 (tcp/udp)	https://docs.syncthing.net/users/firewall.html	`shared_roles/syncthing/defaults/main.yml`	внутренний
21027 (udp)	https://docs.syncthing.net/users/firewall.html		внутренний
Dnsmasq			
53	`dnsmasq`		внутренний
Операционная система			
22	Порт SSH. Используется для развертывания и администрирования ППО. **ВНИМАНИЕ!** Возможность использования данного порта определяется документацией СЗИ от НСД		внутренний

ПРИМЕЧАНИЕ. Рекомендуется запретить доступ к ППО привилегированных пользователей из-за пределов контролируемой зоны, запретив доступ к Консоли администратора ПБ. Также при необходимости можно запретить доступ к остальным веб-консолям. Для этого следует разрешить трафик только по требуемым URL-адресам в соответствии с п. 3.10.1.

3.10.7.2. Настройка ОС Debian и ОС Ubuntu

ПРИМЕЧАНИЕ. Настройка ОС Astra Linux, ОС Альт и РЕД ОС осуществляется в соответствии с эксплуатационной документацией на данные ОС.

3.10.7.2.1 Для затруднения возможностей сбора информации о системе необходимо исключить метки времени из заголовков TCP пакетов, выполнив следующие действия:

3.10.7.2.1.1 В конфигурационный файл `/etc/sysctl.conf` добавить строку:

```
net.ipv4.tcp_timestamps = 0
```

3.10.7.2.1.2 Применить конфигурацию, выполнив команду:

```
sysctl -p /etc/sysctl.conf
```

3.10.7.2.1.3 Проверить корректность конфигурации, выполнив команду:

```
sysctl -a | grep net.ipv4.tcp_timestamps
```

Если настройки заданы правильно, должно быть выведено значение:

```
net.ipv4.tcp_timestamps = 0
```

3.10.7.2.2 Настройка запрета SSH доступа к серверам приложений по логину и паролю.

3.10.7.2.2.1 В конфигурационном файле `/etc/ssh/sshd_config` задать следующие значения параметров:

```
PasswordAuthentication no
PubkeyAuthentication yes
AuthenticationMethods publickey
```

3.10.7.2.2.2 Перезапустить службу `sshd` с помощью команды:

```
sudo service sshd reload
```

3.10.7.2.3 Настройка минимальной сложности пароля.

Настройка сложности пароля осуществляется в конфигурационном файле `/etc/security/pwquality.conf`. Рекомендуется задать следующие значения параметров:

– минимальная длина пароля:

```
minlen = 8
```

– алфавит пароля (минимальное количество используемых классов символов):

```
minclass = 4
```

– максимальная длина последовательности символов (abcd, 12345 и т.п.):

```
maxsequence = 3
```

– максимальное число идущих подряд одинаковых символов:

```
maxrepeat = 3
```

ПРИМЕЧАНИЕ. Настройка конфигурационных файлов должна выполняться на каждой ноде кластера.

3.10.8. Самостоятельная установка необходимых пакетов на серверы приложений, серверы БД и контент-серверы

3.10.8.1. Получить список необходимых пакетов.

Перечень необходимых пакетов, которые должны быть установлены на серверы приложений, серверы БД и контент-серверы, задан в файле `play-managed-node-prerequisites.yml`, находящемся в каталоге со сценариями установки ППО и имеющем следующую структуру:

```
...
    - name: install requirements on <операционная система>
...
        - name: <операционная система> | DB node | Install OS packages
          loop:
            <перечень пакетов сервера БД>
        - name: <операционная система> | APP node | Install OS
packages
          loop:
            <перечень пакетов сервера приложений>
```

В секции `name: install requirements on <операционная система>` задается перечень пакетов для указанной ОС. Данная секция содержит 2 подсекции, в которых задается перечень пакетов для сервера приложений, сервера БД и контент-сервера.

В подсекции `name: <операционная система> | DB node | Install OS packages` задается перечень пакетов для сервера БД.

В подсекции `<операционная система> | APP node | Install OS packages` задается перечень пакетов для сервера приложений и контент-сервера.

Пример перечня пакетов для сервера приложений, сервера БД и контент-сервера, функционирующих под управлением ОС Astra Linux:

```
...
  - name: Install requirements on Astra Linux
      block:
        - debug:
            msg: Install requirements on Astra Linux

        - name: Astra Linux | Add en_US.UTF-8 locale
          locale_gen:
            name: en_US.UTF-8
            state: present

        - name: Astra Linux | DB node | Install OS packages
          apt:
            name: "{{ item }}"
            state: present
            update_cache: true
          loop:
            - acl
            - python3-apt
            - python3-distutils
            - jq
            - unzip
            - curl
            - psmisc
            - sgml-base
            - libxml2
            - libxslt1.1
            - ssl-cert
            - xml-core
            - python3-psycopg2
            - xz-utils
            - rsync
          when: node_type == "db" or node_type == "all"

        - name: Astra Linux | APP node | Install OS packages
          apt:
            name: "{{ item }}"
            state: present
            update_cache: true
          loop:
            - acl
            - python3-apt
            - dnsutils
            - net-tools
            - jq
            - curl
            - unzip
            - rsync
            - libpcre3
            - zlib1g
            - psmisc
```

```
            - iptables
            - dnsmasq
            - xz-utils
            - rsync
            - libuser
        when: node_type == "app" or node_type == "all"
    when: ansible_distribution | regex_search('Astra Linux')
```

3.10.8.2. Установить пакеты.

Установка пакетов осуществляется в соответствии с документацией на ОС.

ПРИМЕЧАНИЕ. Установка пакетов должна выполняться для каждой ноды кластера.

3.10.9. Отключение служб SELinux и Firewalld

Для отключения служб SELinux и Firewalld необходимо выполнить следующие действия:

3.10.9.1. В конфигурационном файле `/etc/selinux/config` задать следующее значение параметра `SELINUX`:

```
SELINUX=disabled
```

3.10.9.2. Отключить в ОС межсетевой экран с помощью выполнения следующих команд:

```
systemctl stop firewalld
systemctl disable firewalld
```

3.10.9.3. Перезагрузить ЭВМ с помощью команды:

```
reboot
```

ПРИМЕЧАНИЕ. Отключить службы SELinux и Firewalld необходимо на каждой ноде кластера.

3.10.10. Требования к установке и настройке внешнего балансировщика (на примере Nginx)

Установка и настройка внешнего балансировщика Nginx осуществляются пользователями (системными администраторами) ППО самостоятельно. Внешний балансировщик должен поддерживать проксирование `http` и `tcp`-соединений.

Для проверки возможности проксирования `tcp`-соединений необходимо выполнить проверку корректности конфигурации Nginx с помощью команды:

```
sudo nginx -t
```

В случае отображения сообщения `unknown directive «stream»` требуется добавить поддержку модуля `ngx_stream_module.so`. Для этого необходимо:

– в конфигурационном файле Nginx (файл: `/etc/nginx/nginx.conf`) добавить строку:

```
load_module '/usr/lib64/nginx/modules/ngx_stream_module.so';
```

– перезапустить Nginx с помощью команды:

```
sudo systemctl reload nginx
```

3.10.10.1. Настройка балансировщика для однотенантной конфигурации:

– выделить домен <AC_DOMAIN> для обращения к ППО, например, `acenter.example.ru`;

– выпустить сертификат для своего домена (доменов), например, `acenter.example.ru`;

– добавить dns-запись для своего домена (доменов), например, `acenter.example.ru`;

– в конфигурационном файле внешнего балансировщика добавить обработку своего домена (доменов), например, `acenter.example.ru`. Примеры конфигурационных файлов приведены в каталоге `samples/ac/nginx_external-balancer/conf.d`:

- one-node.conf – пример конфигурационного файла для сервера приложений;
- content-server.conf – пример конфигурационного файла для контент-сервера.

3.10.10.2. Настройка балансировщика для поддержки мультитенантной конфигурации.

В связи с тем, что для каждого тенанта используется отдельный поддомен, необходимо настроить обработку домена и поддоменов на внешнем балансировщике, выполнив следующие действия:

– выделить домен <AC_DOMAIN> для обращения к ППО, например, acenter.example.ru;

– выпустить обычный и wildcard сертификаты для своего домена (доменов), например acenter.example.ru и *.acenter.example.ru;

– добавить dns-запись для своего домена (доменов) и wildcard запись для поддоменов, например acenter.example.ru и *.acenter.example.ru;

– в конфигурационном файле внешнего балансировщика добавить обработку своего домена (доменов) и поддоменов, например acenter.example.ru и *.acenter.example.ru. Примеры конфигурационных файлов приведены в каталоге samples/ac-mt/nginx_external-balancer/conf_stream.d:

- one-node.conf – пример конфигурационного файла для сервера приложений;
- content-server.conf – пример конфигурационного файла для контент-сервера.

3.10.11. Активация (разблокировка) учетной записи пользователя с помощью sql-запроса к БД

Разблокировка учетных записей пользователей ППО осуществляется Администратором учетных записей с помощью Консоли администратора ПБ. Однако учетная запись Администратора учетных записей также может быть заблокирована (например, при длительной неактивности Администратора учетных записей).

В этом случае для разблокировки учетной записи необходимо выполнить следующие действия:

3.10.11.1. Подключится к БД ПБ (`auth`) с помощью команды:

```
psql -U auth -h <ip-адрес сервера БД> -d auth
```

Например:

```
psql -U auth -h 192.168.0.107 -d auth
```

3.10.11.2. Разблокировать учетную запись пользователя с помощью с sql-запроса:

```
update accounts_users.accounts set is_active=true,
last_activity_at=now() where login='<email пользователя>';
```

Например:

```
update accounts_users.accounts set is_active=true,
last_activity_at=now() where login='admin@omprussia.ru';
```

3.10.12. Действия после сброса устройств к заводским настройкам

Сброс устройства возвращает его к заводским настройкам. После сброса устройств в зависимости от способа их первоначальной установки приложения ППО (приложение «Аврора Центр» и приложение «Аврора Маркет») могут отсутствовать либо быть сброшены до первоначальной версии.

После сброса устройства необходимо выполнить следующие действия:

1) Установить приложения ППО, если после сброса устройства они отсутствуют;

2) Активировать устройство в ПУ в соответствии с документом «Руководство пользователя. Часть 3. Подсистема Платформа управления»;

3) Обновить приложения ППО в соответствии с документом «Руководство пользователя. Часть 7. Приложение «Аврора Центр» для операционной системы Аврора».

3.10.13. Порядок задания адресов (доменных имен) в инвентарном файле inventories/hosts.yml

В инвентарном файле `inventories/hosts.yml` задаются адреса серверов (имена хостов), на которые установлены (будут установлены) компоненты среды функционирования ППО, инфраструктурные компоненты ППО и подсистемы ППО.

Задание адресов (доменных имен) осуществляется посредством их добавления в секцию `hosts`, например:

```
...
        app:
          hosts:
            acenterapp01:
            acenterapp02:
            acenterapp03:
```

Допускается добавление адресов при помощи добавления хостов в группы и дальнейшего переиспользования групп. Например, для Nginx будут заданы адреса из группы `app`, которая заполнена выше:

```
...
    ocs:
      children:
        app:
          hosts:
            acenterapp01:
            acenterapp02:
            acenterapp03:
        nginx:
          children:
            app:
```

Допускается смешанное задание адресов посредством их добавления в секцию `hosts`, а также посредством добавления хостов в группы и дальнейшего переиспользования групп. Например, для Nginx будут заданы адреса из группы `app`, которая заполнена выше, и адреса из секции `hosts`:

```
...
    ocs:
      children:
        app:
          hosts:
            acenterapp01:
            acenterapp02:
            acenterapp03:
        nginx:
          children:
            app:
          hosts:
            acenterapp04:
            acenterapp05:
```

При необходимости установки на хост определенных подсистем ППО потребуется после адреса хоста добавить параметр `subsystems` с перечнем подсистем, например:

```
...
        app:
          hosts:
            acenterapp01:
              subsystems: auth
            acenterapp02:
              subsystems: emm
            acenterapp03:
              subsystems: appstore,pkgrepo
```

Конфигурационный файл сценария установки среды функционирования ППО и инфраструктурных компонентов ППО на 1 ЭВМ с доменным именем `ocs-app.local` имеет следующий вид:

```
all:
  children:
    ocs:
      children:
        app:
          hosts:
            ocs-app.local:
```

```
        content:
          hosts:
        postgresql:
          hosts:
            ocs-app.local:
        nginx:
          children:
            app:
            content:
        consul:
          children:
            consul_servers:
              children:
                app:
            consul_agents:
              children:
        consul_content:
          children:
            content:
        consul_template:
          children:
            app:
            content:
        redpanda:
          children:
            app:
        valkey:
          children:
            app:
              children:
                app:
        sentinel:
          children:
            app:
        syncthing:
          children:
            app:
```

Примеры файлов `hosts.yml` для однонодовой и кластерной конфигураций приведены в каталоге `samples/ac/inventories/` (или в каталоге `samples/ac/inventories/`).

Описание параметров инвентарного файла `inventories/hosts.yml` приведено в п. 12.1.1.

3.10.14. Порядок настройки срока хранения событий безопасности

Срок хранения событий безопасности задается в поле `retention` таблицы `partman.part_config events` БД ПБ (`auth`).

Для просмотра и изменения срока хранения необходимо выполнить следующую последовательность действий:

3.10.14.1. Подключиться к БД ПБ (`auth`) с помощью команды:

```
psql -U auth -h <ip-адрес сервера БД> -d auth
```

Например:

```
psql -U auth -h 192.168.0.107 -d auth
```

3.10.14.2. Просмотреть текущее значение срока хранения с помощью скрипта:

```
select retention from partman.part_config where parent_table =
'audit.audit_events';
```

3.10.14.3. Изменить срок хранения с помощью скрипта:

```
UPDATE partman.part_config SET retention = '<количество дней> days'
where parent_table = 'audit.audit_events';
```

Например:

```
UPDATE partman.part_config SET retention = '90 days' where
parent_table = 'audit.audit_events';
```

3.10.15. Порядок настройки срока хранения событий из журнала сообщений устройств

Срок хранения событий из журнала сообщений устройств задается в поле `retention` таблицы `partman.part_config events` ПУ (`emm`).

Для просмотра и изменения срока хранения необходимо выполнить следующую последовательность действий:

3.10.15.1. Подключиться к БД ПУ (`emm`) с помощью команды:

```
psql -U emm -h <ip-адрес сервера БД> -d emm
```

Например:

```
psql -U emm -h 192.168.0.107 -d emm
```

3.10.15.2. Просмотреть текущее значение срока хранения с помощью скрипта:

```
select retention from partman.part_config where parent_table =
'journal.client_journal_messages';
```

3.10.15.3. Изменить срок хранения с помощью скрипта:

```
UPDATE partman.part_config SET retention = '<количество дней> days'
where parent_table = 'journal.client_journal_messages';
```

Например:

```
UPDATE partman.part_config SET retention = '180 days' where
parent_table = 'journal.client_journal_messages’;
```

3.10.16. Порядок настройки ППО для его установки на различные окружения

Сценарии установки позволяют выполнить настройку и установку ППО, а также компонентов среды функционирования ППО для нескольких различных окружений, выполнив следующие действия:

3.10.16.1. Перейти в каталог `/install-<версия ППО>/install-ac-mt/`

3.10.16.2. Создать инвентарный файл `hosts.yml` по пути: `inventories/<название окружения>/hosts.yml`.

Описание параметров инвентарного файла `hosts.yml` приведено в п. 12.1.1.

3.10.16.3. Создать каталог `config/environments/<название окружения>/`, создать в данном каталоге требуемые конфигурационные файлы с учетом их расположения в каталоге `config` и задать требуемые значения параметров.

Более подробная информация по работе с конфигурационными файлами окружения приведена в п. 13.2.8.

3.10.16.4. Выполнить установку компонентов среды функционирования ППО и ППО в соответствии с подразделом 3.6 для заданного окружения, указав в командах установки путь к инвентарному файлу и имя окружения.

Примеры команд:

– команда установки всех пакетов на все серверы (на все серверы приложений, серверы БД и контент-серверы независимо от их типа):

```
ansible-playbook -i inventories/<название окружения>/hosts.yml play-managed-node-prerequisites.yml -vv -u <имя пользователя>
```

– команда установки компонентов среды функционирования ППО и инфраструктурных компонентов ППО:

```
ANSIBLE_USER="<имя пользователя>" ./deploy-infra.sh --env "<название окружения>"
```

– команда установки ППО:

```
ANSIBLE_USER="<имя пользователя>" ./deploy-ac.sh --env "<название окружения>"
```

3.10.17. Удаление персональных данных из учетной записи пользователя, персональных данных контактного лица организации и персональных данных контактного лица проекта

3.10.17.1. Для удаления персональных данных из учетной записи пользователя необходимо выполнить следующие действия:

– подключиться к БД ПБ (`auth`) с помощью следующей команды:

```
psql -U auth -h <ip-адрес сервера БД> -d auth
```

Например:

```
psql -U auth -h 192.168.0.107 -d auth
```

– выполнить `sql`-запрос:

```
update accounts_users.accounts set login='', last_name='', first_name='', patronymic='' where login='<email пользователя>';
```

Например:

```
update accounts_users.accounts set login='', last_name='', first_name='', patronymic='' where login='ivanov@omprussia.ru';
```

3.10.17.2. Для удаления персональных данных контактного лица организации необходимо выполнить следующие действия:

– выполнить архивацию контактного лица, если контактное лицо не заархивировано;

– подключиться к БД ПУТ (mt) с помощью следующей команды:

```
psql -U mt -h <ip-адрес сервера БД> -d mt
```

Например:

```
psql -U mt -h 192.168.0.107 -d mt
```

– выполнить sql-запрос:

```
UPDATE organizations.contact_persons
SET first_name='Deleted',
    last_name='Deleted',
    patronymic=NULL,
    comment=NULL,
    phone='112',
    email=id::text || '@example.com'
WHERE deleted_at IS NOT NULL AND email='<email контактного лица>';
```

Например:

```
UPDATE organizations.contact_persons
SET first_name='Deleted',
    last_name='Deleted',
    patronymic=NULL,
    comment=NULL,
    phone='112',
    email=id::text || '@example.com'
WHERE deleted_at IS NOT NULL AND email='ivanov@omprussia.ru';
```

3.10.17.3. Для удаления персональных данных пользователей устройств необходимо выполнить следующие действия:

– зайти в карточку пользователя устройства и выполнить архивацию пользователя, если пользователь не заархивирован;

– подключиться к БД ПУ (emm) с помощью следующей команды:

```
psql -U emm -h <ip-адрес сервера БД> -d emm
```

Например:

```
psql -U emm -h 192.168.0.107 -d emm
```

– выполнить sql-запросы:

```
UPDATE users_service.users
SET first_name   = 'Deleted',
    last_name    = 'Deleted',
    patronymic   = NULL,
    job_title    = NULL,
    phone_number = NULL,
    email        = id::text || '@example.com'
WHERE email = '<email пользователя МУ>' AND deleted_at IS NOT NULL;

UPDATE users_service.users_read_model
SET first_name   = 'Deleted',
    last_name    = 'Deleted',
    patronymic   = NULL,
    job_title    = NULL,
    phone_number = NULL,
    email        = id::text || '@example.com'
WHERE email = '<email пользователя МУ>' AND deleted_at IS NOT NULL;
```

Например:

```
UPDATE users_service.users
SET first_name   = 'Deleted',
    last_name    = 'Deleted',
    patronymic   = NULL,
    job_title    = NULL,
    phone_number = NULL,
    email        = id::text || '@example.com'
WHERE email = 'ivanov@omprussia.ru' AND deleted_at IS NOT NULL;

UPDATE users_service.users_read_model
SET first_name   = 'Deleted',
    last_name    = 'Deleted',
    patronymic   = NULL,
    job_title    = NULL,
    phone_number = NULL,
    email        = id::text || '@example.com'
WHERE email = 'ivanov@omprussia.ru' AND deleted_at IS NOT NULL;
```

3.10.17.4. Для удаления персональных данных контактного лица проекта ПСУ необходимо выполнить следующие действия:

– подключиться к БД ПСУ (`push`) с помощью следующей команды:

```
psql -U push -h <ip-адрес сервера БД> -d push
```

Например:

```
psql -U push -h 192.168.0.107 -d push
```

– выполнить sql-запрос:

```
UPDATE main.contact_persons
SET
    first_name='Deleted',
    last_name='Deleted',
    patronymic='Deleted',
    position='Deleted',
    phone='112',
    email=id::text || '@example.com'
WHERE email='<email контактного лица>';
```

Например:

```
UPDATE main.contact_persons
SET
    first_name='Deleted',
    last_name='Deleted',
    patronymic='Deleted',
    position='Deleted',
    phone='112',
    email=id::text || '@example.com'
WHERE email='ivanov@omprussia.ru';
```

3.10.18. Сброс пароля учетной записи

В случае утери пароля от учетной записи с ролью Администратор учетных записей и невозможности его восстановления штатным способом (например, если в ППО была только 1 учетная запись с указанной ролью), необходимо выполнить следующие действия для сброса пароля учетной записи:

– подключиться к БД ПБ (auth) с помощью команды:

```
psql -U auth -h <ip-адрес сервера БД> -d auth
```

Например:

```
psql -U auth -h 192.168.0.107 -d auth
```

– в файле samples/sql/activate_user_account.sql, расположенном в каталоге со сценариями установки ППО, задать логин учетной записи в параметре accountLogin, например:

```
accountLogin text := 'admin@omprussia.ru'
```

– скопировать содержимое файла activate_user_account.sql в консоль и выполнить скрипт, нажав клавишу «Enter».

После выполнения указанных действия пароль будет иметь значение «admin».

3.10.19. Восстановление учетной записи пользователя тенанта в случае ее удаления

Для восстановления учетной записи пользователя тенанта в случае ее удаления необходимо выполнить следующие действия:

– подключиться к БД ПБ (auth) с помощью следующей команды:

```
psql -U auth -h <ip-адрес сервера БД> -d auth
```

Например:

```
psql -U auth -h 192.168.0.107 -d auth
```

– в файле samples/sql/create_tenant_default_user_account.sql, находящемся в каталоге со сценариями установки ППО, задать логин учетной записи (параметр: accountLogin) и код тенанта (параметр: accountTenantCode), например:

```
accountLogin text := 'admin@omprussia.ru';
accountTenantCode text := 'default';
```

ПРИМЕЧАНИЕ. Код тенанта доступен в карточке тенанта;

– скопировать в консоль содержимое файла create_tenant_default_user_account.sql и выполнить скрипт, нажав клавишу «Enter».

3.10.20. Настройка включения/отключения регистрации событий

Настройка регистрации событий осуществляется в конфигурационных файлах шлюзов доступа `endpoints.yml`, которые располагаются на сервере приложений ППО в каталоге:

```
/var/ocs/config/subsystems/<название подсистемы>/applications/<название шлюза доступа>/endpoints.yml
```

Например:

```
 /var/ocs/config/subsystems/auth/applications/ocs-auth-admin-api-gw/endpoints.yml
```

либо в каталоге со сценариями установки ППО:

```
config/subsystems/<название подсистемы>/applications/<название шлюза доступа>/endpoints.yml
```

Например:

```
config/subsystems/auth/applications/ocs-auth-admin-api-gw/endpoints.yml
```

Для отключения/включения регистрации события для функции ППО (эндпоинта) необходимо в секции требуемого эндпоинта закомментировать/раскомментировать секцию `audit`, например:

```
- endpoint: /api/devices/{id}/operations/next
  method: GET
  backends:
    - url_pattern: /v1/devices/{id}/operations/next
      host: ocs-emm-dispatcher-api.${domain}
      forward_error: true
  auth:
    scope: operation:read
  permissions:
    resource_type: operation
    action: read
    context_field_map:
      device.id: request.params.id
#  audit:
#    field_map:
#      action: get
#      object_id: response.body.id
#      object_label: response.body.type
#      object_type: deviceOperation
```

Далее в зависимости от типа конфигурационного файла выполнить переустановку конфигурационного файла или перезапуск сервиса. Подробная информация об управлении настройками сервисов ППО приведена в подразделе 4.3.

3.10.21. Настройка брендирования ППО

Для добавления логотипа компании и выбора цветовой схемы графического интерфейса ППО необходимо в конфигурационном файле `config/internal.yml` задать следующие параметры:

– `brandingLogoUrl` – ссылка на изображение, либо изображение логотипа в формате base64 (Изображение будет размером 160x32 точек);

– `brandingLogoAlt` – текст, который будет отображаться при наведении на изображение логотипа;

– `theme` – описание цветовой схемы Material UI: цвета кнопок и текста, а также прочие настройки (см. https://mui.com/material-ui/customization/palette/). Не рекомендуется менять шрифты и их размеры, т.к. неправильные значения могут нарушить корректность отображения интерфейса.

Например:

```
brandingLogoUrl: "
...",
brandingLogoAlt: "test branding text",
theme:
  palette:
    primary:
      main: "#ff4200"
      light: "#ff4200"
      dark: "#ff4200"
      contrastText: "#333333"
    secondary:
      main: "#ff4200"
      light: "#ff4200"
      dark: "#ff4200"
      contrastText: '#333333'
```

3.10.22. Переключение трафика между ЦОД (failover/switchover)

Сервисы ППО поддерживают одновременную работу в двух ЦОД, однако СУБД PostgreSQL может обрабатывать транзакции на запись (Active) только на одной площадке.

ВНИМАНИЕ! Переключение PostgreSQL на резервный ЦОД выполняется в ручном режиме. Важно не допустить одновременного наличия двух Primary-кластеров. Если перевести Standby-кластер в резервном ЦОД в режим Primary, предварительно не остановив СУБД в Основном ЦОД, произойдет «Split-Brain» (рассинхронизация данных). Потеря данных при переключении БД между ЦОД будет равна лагу репликации, а также время, затраченное администратором на выполнение команд переключения.

Для переключения трафика с основного ЦОД на резервный рекомендуется выполнить следующие действия:

3.10.22.1. Перевести кластер БД в резервном ЦОД из состояния StandBy в Primary.

Для этого необходимо выполнить следующие действия:

3.10.22.1.1 Остановить Primary-кластер Postgres в Основном ЦОД с помощью команды:

```
systemctl stop patroni
```

3.10.22.1.2 Остановить Patroni на каждом сервере БД в Основном ЦОД и убедиться в отсутствии активных процессов PostgreSQL:

```
ps -ef | grep postgres
```

3.10.22.1.3 Перевести Standby-кластер в Резервном ЦОД в режим Primary. На одном из серверов БД в Резервном ЦОД удалить директиву `standby_cluster` из динамической конфигурации Patroni:

```
patronictl -c /etc/patroni.yml edit-config --force --set
standby_cluster=''
```

3.10.22.1.4 Проверить статуса нового кластера Postgres. Убедиться, что один из серверов резервного ЦОД успешно принял на себя роль Leader:

```
patronictl -c /etc/patroni.yml list
```

3.10.22.2. Перевести входящий трафик на внешнем балансировщике из основного ЦОД в резервный.

3.10.22.3. После восстановления кластера БД в основном ЦОД, можно перевести кластер БД в основном ЦОД в состояние `StandBy`.

ВНИМАНИЕ! Когда аварийный (ранее Основной) ЦОД восстановит работу, его кластер баз данных все еще будет находиться в роли Primary. Не запускать на нем Patroni без предварительной настройки, чтобы избежать конфликта двух Primary-узлов.

Настроить каскадную репликацию, добавив в динамическую конфигурацию восстанавливаемого кластера параметры `standby_cluster`, указав в качестве источника адреса серверов нового основного кластера. Добавить в динамическую конфигурацию настройки `standby_cluster` с помощью команды:

```
patronictl -c /etc/patroni.yml edit-config --force   \
--set standby_cluster.host='<acenter-db-dc2-03,acenter-db-dc2-04>' \
--set standby_cluster.port=5432                                    \
--set standby_cluster.create_replica_methods='- basebackup'
```

где `acenter-db-dc2-03, acenter-db-dc2-04`– `virtual IP address` сервера, с которого будет настроена репликация данных.

Затем проверить конфигурации с помощью команды:

```
patronictl -c /etc/patroni.yml show-config
```

3.10.23. Рекомендуемые действия при сильной рассинхронизации или зависании репликаци

Если реплики разошлись, старый кластер сильно отстал или его запуск зависает на сообщении «`waiting for standby_leader to bootstrap`», требуется полная переинициализация. Для этого выполнить следующие действия:

3.10.23.1. Остановить службу Patroni на проблемном сервере (Standby Leader), выполнив команду:

```
systemctl stop patroni
```

3.10.23.2. Полностью удалить директорию с данными PostgreSQL (путь зависит от версии, пример для PG 16):

```
rm -rf /var/lib/postgres/16/main
```

3.10.23.3. Удалить старый кластер из конфигурации Patroni:

```
patronictl -c /etc/patroni.yml remove <standby-cluster>
```

3.10.23.4. В файл `/etc/patroni.yml` в секцию `dcs` добавить блок `standby_cluster`:

```
standby_cluster:
  host: <primary_cluster_host>
  port: 5432
  primary_slot_name: patroni
  create_replica_methods:
    - basebackup
```

3.10.23.5. Запустить службу Patroni с помощью команды:

```
systemctl start patroni
```

3.10.23.6. Отследить по логам Patroni процесс повторной инициализации через `basebackup`. После завершения проверить роли узлов командой:

```
patronictl -c /etc/patroni.yml list
```

3.10.24. Настройка адреса проверки подключения устройств к сети

По умолчанию, для проверки доступа к сети, устройства обращаются к публичному адресу http:// ipv4.omprussia.ru, который доступен из сети Интернет.

В случае использования устройств в закрытой сети (сеть без доступа к сети Интернет), необходимо данный адрес заменить на адрес, который будет доступен из закрытой сети и зарегистрирован на сервере DNS, находящемся в закрытой сети.

Для того, чтобы устройства могли выполнять проверку доступа к сети необходимо:

3.10.24.1. В закрытой сети настроить сервер Nginx со следующими параметрами:

```
server {
    listen 80;

    server_name <имя сервера>;


  access_log /var/log/nginx/access-ipv4.omprussia.ru.log main;
  error_log /var/log/nginx/error-ipv4.omprussia.ru.log;


  client_max_body_size 1K;

    location / {
        return 404;
    }

    location = /return_204 {
        return 204;
        access_log /var/log/nginx/ipv4_return_204_access.log;

    }
}
```

ПРИМЕЧАНИЕ. Указанные настройки могут быть выполнены либо на внешнем балансировщике, либо на отдельном сервере. На внутреннем балансировщике указанные настройки выполнять запрещено.

3.10.24.2. В параметрах `networkCheckSettings.ipv4Url` и `networkCheckSettings.ipv6Url` конфигурационного файла ПУ `install-<версия ППО>/install-ac-mt/config/subsystems/emm/config.yml` задать адрес сервера Nginx, к которому будут обращаться устройства. Данный адрес будет автоматически распространяться в настройки каждого устройства с ОС Аврора, подключенного к ПУ. Изменение настроек на устройстве будет отражено в Консоли администратора ПУ в карточке устройства в разделе «Состояние».

3.10.24.3. Переустановить конфигурационные файлы сервиса `ocs-emm-state-manager-api` с помощью команды:

```
ANSIBLE_USER="<имя пользователя>" ./deploy-ac.sh --subsystems emm --apps ocs-emm-state-manager-api --action config
```

3.10.25. Настройка подключения репозитория ОС Linux

Репозиторий ОС Linux используется для установки и обновления программного обеспечения на устройствах с ОС семейства Linux.

Устройства с репозиторием могут взаимодействовать как напрямую, так и через контент-сервер. Взаимодействие с репозиторием через контент-сервер требуется в тех случаях, когда необходим аутентифицированный доступ к репозиторию.

Каким образом будет осуществляться взаимодействие с репозиторием зависит от параметров подключения, указанного в правиле политики «Конфигурация репозиториев/Подключение системных репозиториев». Порядок работы с политиками приведен в документе «Руководство пользователя. Часть 3. Подсистема Платформа управления».

В случае, если взаимодействие с репозиторием осуществляется через контент-сервер, то необходимо в секции `config.publicUris.pkgrepo.linuxRepoAddresses` конфигурационного файла `config/config.yml.j2` задать его адрес, имя и тип, например:

```
pkgrepo:
# Linux-репозитории, содержимое которых предоставляется через контент серверы ППО
    linuxRepoAddresses:
        - name: "test-repo" # Уникальное имя репозитория (используется только символы, разрешенные для URL без кодировки)
          type: "rpm" # Тип репозитория в зависимости от пакетов: rpm, deb, flatpak
          address: https://repo.linux.ru/ # Адрес репозитория
```

ВНИМАНИЕ! Имя репозитория может содержать только цифры, заглавные и строчные буквы (кириллица не допускается).

3.10.26. Настройка защищенного TLS/SSL соединения с БД

По умолчанию для подключения подсистем ППО к БД используется незащищеное TCP-соединение.

3.10.26.1. Настройка защищенного TLS/SSL соединения с помощью скриптов установки ППО

При установке СУБД Postgres с помощью скриптов из состава дистрибутива Аврора Центр имеется возможность автоматически настроить защищенное TLS/SSL соединения с использованием самоподписанных сертификатов.

Для этого необходимо выполнить следующие действия:

1) Сгенерировать CA-ключ и CA-сертификат, которым будет подписан сертификат сервера БД, например:

```
openssl genrsa -out ca_key.pem -aes256 # Запомнить пароль к ключу
openssl req -new -x509 -nodes -days 365000 -key ca_key.pem -out ca-
cert.pem
```

2) Добавить CA-ключ и CA-сертификат в конфигурационный файл `config/vars/_vars.yml`:

```
ca_crt: |
  -----BEGIN CERTIFICATE-----
  …
  -----END CERTIFICATE-----
ca_key:
  -----BEGIN RSA PRIVATE KEY-----
  …
  -----END RSA PRIVATE KEY-----
ca_key_passphrase:
pg_with_tls: true
pg_auth_method: scram-sha-256
```

3) Настроить подключение сервисов к БД в защищенном режиме, задав переменные в конфигурационном файле `config/vars/_vars.yml`:

```
postgresql:
  # допустимые значения:
  #[disable|require|verify-ca|verify-full]
  sslmode: require
```

ПРИМЕЧАНИЕ. Сертификат сервера БД будет автоматически сгенерирован и указан в конфигурационных файлах при установке ППО.

4) Продолжить настройку и установку ППО в соответствии с подразделами 3.5 и 3.6

3.10.26.2. Настройка защищенного TLS/SSL соединения при самостоятельной установке СУБД:

1) Включить защищенное TLS/SSL соединение (`ssl = 'on'`) и указать путь к корневому сертификату (`ca.crt`), сертификату и ключу СУБД (`postgresql.crt` и `postgresql.key`) в конфигурационном файле СУБД `postgresql.conf`:

```
ssl = 'on'
ssl_ca_file = '/var/lib/pgsql/14/data/../ca.crt'
ssl_cert_file = '/var/lib/pgsql/14/data/../postgresql.crt'
ssl_key_file = '/var/lib/pgsql/14/data/../postgresql.key'
```

2) Настроить подключение сервисов к БД в защищенном режиме, задав переменные в конфигурационном файле `config/vars/_vars.yml`:

```
postgresql:
  # допустимые значения:
  #[disable|require|verify-ca|verify-full]
  sslmode: require
```

3) Продолжить настройку и установку ППО в соответствии с подразделами 3.5 и 3.6.

3.10.27. Добавление корневого сертификата в доверенные для настройки защищенного TLS/SSL соединения

3.10.27.1. Установка корневого сертификата для ОС на базе RHEL (РЕД ОС, Альт)

Для работы защищенного TLS/SSL соединения необходимо выполнить следующие действия:

3.10.27.1.1 Разместить доверенный сертификат в каталоге `/etc/pki/tls/certs/`

ВНИМАНИЕ! Если сертификат получен в формате DER, то его необходимо переконвертировать в формат PEM с помощью команды:

```
openssl x509 -inform der -in /path/to/cert/test.cer -out test.pem
```

3.10.27.1.2 Перезапустить сервисы ППО с помощью команды:

```
systemctl restart ocs-*
```

3.10.27.2. Настройка защищенного TLS/SSL соединения для ОС на базе Debian (Astra Linux, Ubuntu)

Для работы защищенного TLS/SSL соединения необходимо выполнить следующие действия:

3.10.27.2.1 Разместить доверенный сертификат в каталоге `/usr/local/share/ca-certificates/`

ВНИМАНИЕ! Если сертификат получен в формате DER, то его необходимо переконвертировать в формат PEM с помощью команды:

```
openssl x509 -inform der -in /path/to/cert/test.cer -out test.pem
```

3.10.27.2.2 Обновить корневые сертификаты с помощью команды:

```
update-ca-certificates -v
```

3.10.27.2.3 Перезапустить сервисы ППО с помощью команды:

```
systemctl restart ocs-*
```

3.10.28. Настройка взаимодействия ПУ и ПСУ при использовании GOST TLS на внешнем балансировщике

ППО не поддерживает отправку push-уведомлений при использовании защищенного протокола GOST TLS на внешнем балансировщике. Если ПСУ расположена на одном сервере с другими подсистемами, то возможно настроить отправку push-уведомлений, минуя шлюз с GOST TLS, сразу на ноды, которые расположены за ним.

Для настройки взаимодействия необходимо выполнить следующие действия:

3.10.28.1. Изменить адрес, который используется для обращения ПУ к ПСУ, с публичного на внутренний, заменив значение параметра `systemUris` в конфигурационном файле `/var/ocs/config/subsystems/emm/config.yml` с:

```
systemUris:
  push:
    publicAddress: "${publicUris.push.publicAddress}"
```

на:

```
systemUris:
  push:
    publicAddress: "http://ocs-push-public-api-gw.${domain}:8007"
```

3.10.28.2. Задать адрес для получения ПУ авторизационных токенов для взаимодействия с ПСУ, добавив в конфигурационный файл сервиса «`ocs-emm-egress-api-gw`» (`/var/ocs/config/subsystems/emm/applications/ocs-emm-egress-api-gw/endpoints.yml`) параметр «`client_assertion_audience`»:

```
- endpoint: /push/public/api/v1/projects/{project_id}/messages
  method: POST
  backends:
    - url_pattern: /api/v1/projects/{project_id}/messages
      host: ${systemUris.push.publicAddress}
      encoding: no-op
      forward_error: true
      client_credentials:
        auth_style: private_key_jwt
        client_assertion_audience:
"${publicUris.auth.publicUri}${tokenEndpoint}"      # Добавляемый
параметр
        client_id: ${pushNotificationSystem.clientId}
        audience: ${pushNotificationSystem.audience}
        ...
        token_url: ${pushNotificationSystem.tokenURL}
  output_encoding: no-op
```

3.10.28.3. Указать адрес для получения авторизационных токенов в настройках интеграции с Сервисом уведомлений Аврора. В Консоли администратора ПУ перейти в подраздел «Настройки» раздела «Администрирование». Внизу подблока «Сервис

уведомлений Аврора» нажать «Редактировать настройки Сервиса уведомлений Аврора». В открывшемся окне в поле «Token URL» ввести:

```
http://ocs-push-public-api-gw.${domain}:8007/auth/public/oauth2/token
```

Сохранить изменения.

3.10.28.4. Перезапустить сервис `ocs-emm-egress-api-gw` с помощью команды:

```
systemctl restart ocs-emm-egress-api-gw*
```

3.10.29. Настройка передачи журнала аудита в SIEM-систему

ППО поддерживает возможность дублирования событий аудита во внешнюю SIEM-систему.

Для передачи событий аудита необходимо в конфигурационном файле сценариев установки `config/subsystems/auth/config.yml` раскомментировать блок `cef` и задать следующие параметры:

- `enabled` – включить интеграцию с SIEM-системой;
- `address` – адрес SIEM-системы;
- `port` – порт SIEM-системы;
- `fieldMap` – сопоставление полей (маппинг) событий аудита ППО.

Поле «fieldMap» является набором пар «ключ - значение» со следующими параметрами:

- `from` – название поля из журнала аудита ППО;
- `to` – название «extension» поля формата CEF;
- `default` – название «extension» поля формата CEF, если в журнале аудита ППО выбранное значение отсутствует или является «null».

Пример раздела «cef»:

```
cef:
  enabled: true
  address: "localhost"
  port: 514
```

```
  proto: "tcp"
  tag: "audit-api"
  fieldMap:
    - from: "action"                          # Действие, совершенное
над объектом
      to: "act"
      default: ""
    - from: "eventTime"                       # Дата и время совершения
события
      to: "start"
      default: ""
    - from: "subjectLogin"                    # Логин инициатора
      to: "suser"
      default: ""
    - from: "subjectId"                       # ID инициатора
      to: "suid"
      default: ""
    - from: "subjectIP"                       # IP адрес инициатора
      to: "dst"
      default: ""
    - from: "requestURL"                      # URL http-запроса
(специальное)
      to: "requestUrl"
      default: ""
    - from: "httpMethod"                      # HTTP-метод
      to: "requestMethod"
      default: ""
    - from: "requestID"                       # ID запроса
      to: "externalId"
      default: ""
    - from: "gatewayId"                       # ID гейтвея
      to: "sourceServiceName"
      default: ""
    - from: "gatewayIP"                       # IP адрес гейтвея
      to: "src"
      default: ""
    - from: "eventOutcome"                    # Результат выполнения
действия (специальное)
      to: "eventOutcome"
      default: ""
```

Возможные значения параметра «from»:

```
"subjectLogin": "Логин инициатора";
"subjectId": "ID инициатора";
"subjectIP": "IP субъекта";
"subjectType": "Тип инициатора";
"objectId": "ID объекта";
"objectLabel": "Название объекта";
"objectType": "Тип объекта";
"action": "Действие, совершенное над объектом";
```

```
"result": "Ожидается HTTP-код";
"endpoint": "Вызванный запрос";
"requestQuery": "Параметры запроса";
"httpMethod": "HTTP-метод";
"gatewayId": "ID гейтвея";
"gatewayIP": "IP гейтвея";
"eventTime": "Дата и время совершения события";
"tenantCode": "Код тенанта";
"requestID": "ID запроса;
"comment": "Комментарий" (на данный момент не используется)
"requestURL": "URL http-запроса" (специальное)
"eventOutcome": "Результат выполнения действия" (специальное)
```

Специальные поля «requestURL» и «eventOutcome» отсутствуют в журнале событий аудита ППО и генерируются автоматически, если заданы в конфигурационном файле. Описание специальных значений параметра «from»:

1) «requestURL» - составное поле, состоящее из значений следующих параметров:

– «config.PublicUris.AC.CommonAddress» конфигурационного файла `config/config.yml.j2`, например «http://ocs-app.local:8009»;

– значения ключа «Endpoint» из журнала аудита ППО, например «/api/policies»;

– значения ключа «RequestQuery» из журнала аудита ППО, например «?limit\=5».

Пример автоматически сгенерированного поля «requestURL»:

```
http://ocs-app.local:8009/api/policies?limit\=5
```

2) «eventOutcome» - поле содержит значение в зависимости от поля «result» журнала аудита ППО и принимает следующие значения:

– «failure» - при значении поля «result» > 399;

– «success» - при значении поля «result» <= 399.

После внесения изменений необходимо переустановить конфигурационные файлы ПБ с помощью команды:

```
ANSIBLE_USER="<имя пользователя>" ./deploy-ac.sh --action config --
subsystems auth
```

3.10.30. Настройка интеграции с репозиторием «РТК-Феникс»

ППО поддерживает возможность проверки RPM-пакетов на наличие уязвимостей с помощью репозитория «РТК-Феникс».

Для этого необходимо задать в конфигурационном файле `config/subsystems/appstore/config.yml` следующие параметры:

– `enabled` – включить интеграцию с репозиторием «РТК-Феникс»;

– `concurrency` – количество одновременных запросов в «РТК-Феникс» проверок пакетов;

– `verificationInterval` – интервал времени между периодическими проверками;

– `baseUrl` – адрес API репозитория «РТК-Феникс»;

– `token` – токен для авторизации.

Например:

```
releaseVerifications:
  rtkFeniks:
    enabled: true
    concurrency: 10
    verificationInterval: "12h"
    conditions:
      OS_ALT_LINUX:
        - "PACKAGE_TYPE_RPM"
    connection:
      baseUrl: http://localhost:12345/
      token: "some_token"
      insecureSkipVerify: false
```

После внесения изменений необходимо переустановить конфигурационные файлы ПМ с помощью команды:

```
ANSIBLE_USER="<имя пользователя>" ./deploy-ac.sh --action config --
subsystems appstore
```

3.10.31. Установка и настройка PXE-сервера

ППО содержит в своем составе PXE-сервер, позволяющий устанавливать ОС по сети.

PXE-сервер состоит из следующих компонентов:

– TFTP-сервер;

– HTTP-сервер.

TFTP-сервер используется для передачи на ЭВМ пользователя загрузчиков и меню iPXE. Для работы необходимо наличие следующих файлов:

– `autoexec.ipxe` – загрузочное меню iPXE;

– `ipxe.efi` – загрузчик меню iPXE для клиентов с EFI/UEFI;

– `undionly.kpxe` – загрузчик меню iPXE для клиентов с BIOS.

HTTP-сервер используется для передачи на ЭВМ пользователя ядра ОС, файлов автоответов, пакетов дистрибутива, а также скриптов.

ПРИМЕЧАНИЕ. Для установки ОС требуется наличие DHCP-сервера, настроенного на работу с PXE-сервером Аврора Центр.

Общий порядок установки ОС по сети:

1) Получение ЭВМ пользователя IP-адреса, адреса boot-сервера (TFTP) и имени файла-загрузчика iPXE от DHCP-сервера;

2) Получение загрузчика iPXE и передача ему управления, загрузка файла с интерактивным меню;

3) Загрузка по сети с HTTP-сервера ядра, временной файловой системы выбранной ОС (`initrd`) и файла автоответов;

4) Передача управления ядру, начало установки ОС на основе информации из файла автоответов;

5) Настройка ОС после окончания ее установки. Настройка ОС может быть выполнена с помощью postinstall-скриптов, которые загружаются с HTTP-сервера. Инициация вызова настройки происходит из файла автоответов в шаге 4.

Для автоматической активации ЭВМ после установки ОС и приложения «Аврора Центр», необходимо:

1) Создать приглашение с нужными условиями и выгрузить его в файл `invite.json`;

2) Добавить `invite.json` в папку `postinstall` (например `pxe/astra_177/postinstall`);

3) Добавить в `postinstall script` скачивание этого файла в папку `/etc/omp-uem-agent/invite.json`

Например:

```
logger -t "$0" "Download invite"
curl -SL "http://{{ ipxe_server_name
}}/ipxe/astra_177/postinstall/invite.json" -o /etc/omp-uem-
agent/invite.json
```

Из данной папки приложение «Аврора Центр» автоматически при старте прочитает `invite.json` и выполнит активацию.

Схема сетевого взаимодействия пользовательской ЭВМ с DHCP- и PXE-серверами приведена на рисунке (Рисунок 19). Схема развертывания PXE-сервера приведена на рисунке (Рисунок 20).

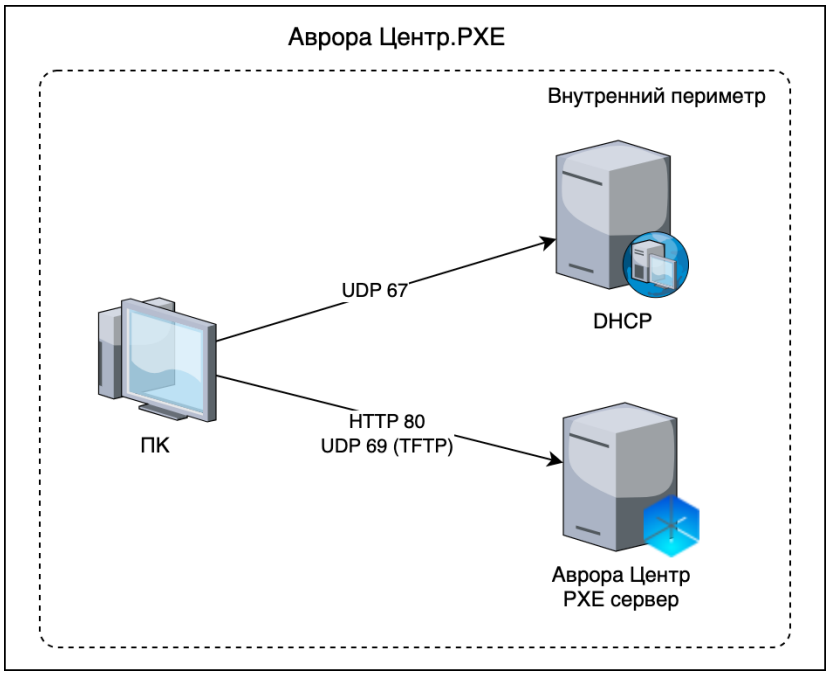


Рисунок 19

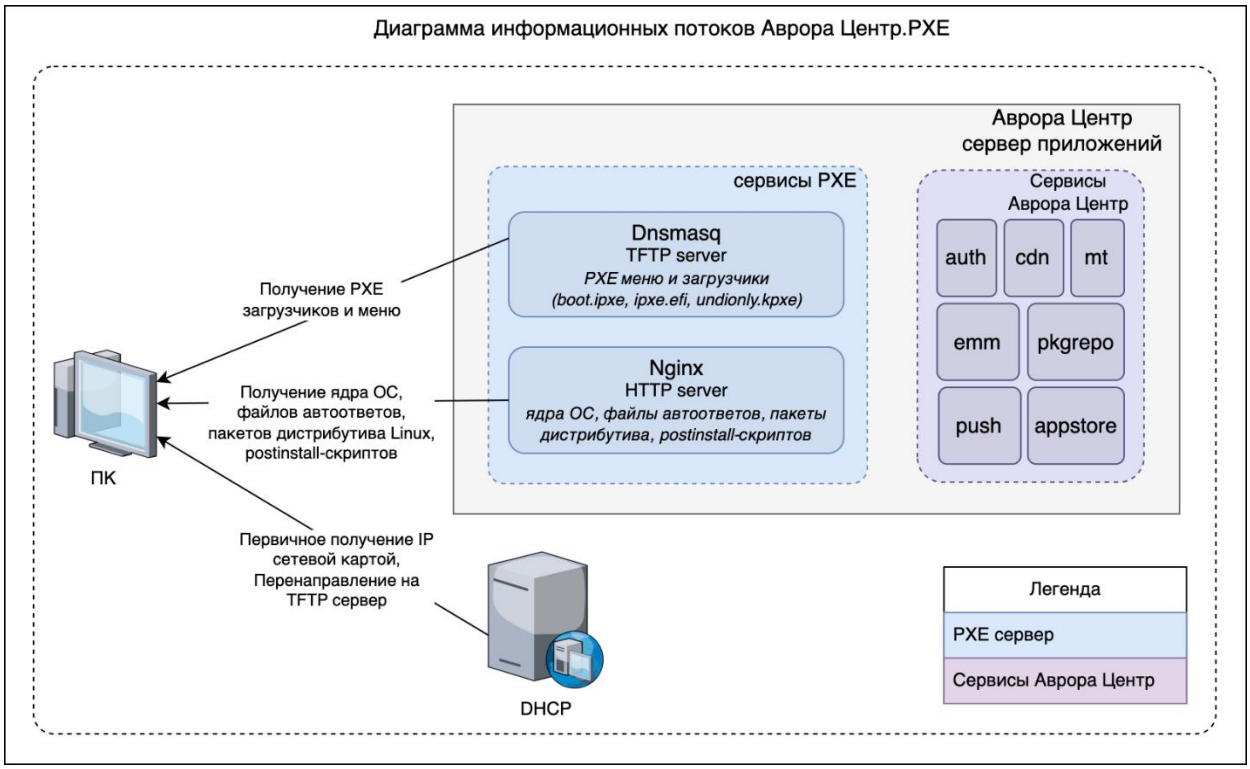


Рисунок 20

ВНИМАНИЕ! Взаимодействие пользовательской ЭВМ с DHCP-сервером и PXE-сервером осуществляется с использованием незащищенных протоколов. Поэтому при наличии угроз со стороны внутреннего нарушителя необходимо предусмотреть меры по защите от данных угроз.

3.10.31.1. Настройка DHCP-сервера

ВНИМАНИЕ! Действия, приведенные в данном пункте, предполагают, что DHCP-сервер уже установлен и настроен на выдачу пользовательским ЭВМ IP-адресов, передачу доменной зоны, адресов серверов имен, шлюза и прочих настроек, и описывают только настройку опций и политик, связанных с PXE.

ВНИМАНИЕ! Настройка DHCP-сервера на коммутаторе или на маршрутизаторе осуществляется в соответствии с эксплуатационной документацией на данное оборудование.

Для развертывания ОС по сети необходимо настроить передачу пользовательским ЭВМ DHCP-сервером опции *TFTP / Boot Server Host Name* (Option 66) и опции *Bootfile Name* (Option 67).

3.10.31.1.1 Настройка Windows Server DHCP

3.10.31.1.1.1 Настройка передачи имени TFTP-сервера (Option 66)

Для настройки передачи имени TFTP-сервера необходимо выполнить следующие действия:

1) В окне управления DHCP нужно выбрать свой контроллер домена, раскрыть пункт «IPv4» и, нажав правой кнопкой на пункт «Scope Options» (для изменения настроек области адресов) или «Server Options» (для применения настроек ко всему серверу), вызвать контекстное меню и выбрать пункт «Configure Options» (Рисунок 21);

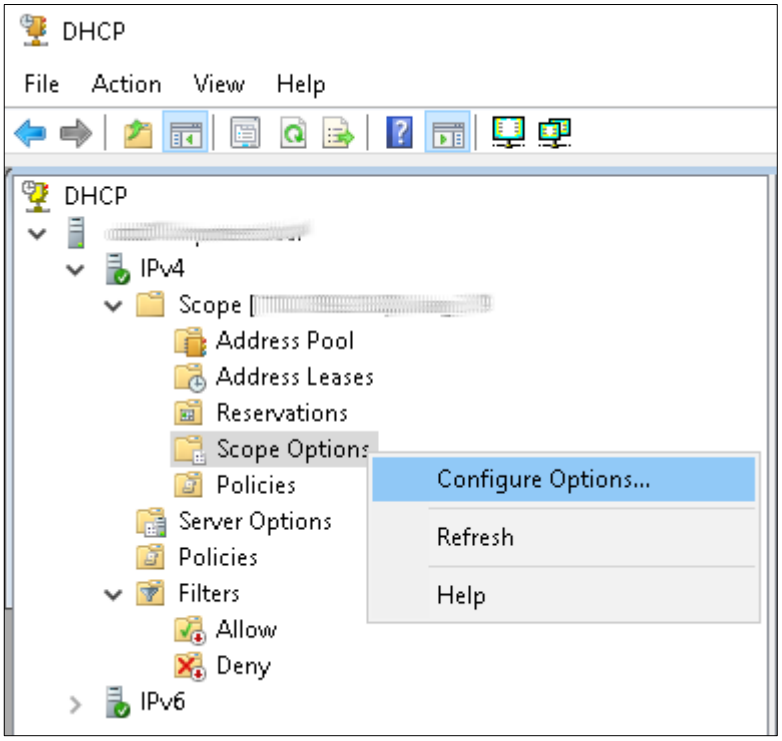


Рисунок 21

2) В открывшемся окне найти опцию 66 (Рисунок 22) и указать доменное имя TFTP-сервера.

Настройку передачи имени TFTP-сервера также можно осуществить с помощью «PowerShell»:

– для «Server Options» выполнить команду:

```
Set-DhcpServerv4OptionValue -OptionID 66 -Value "<доменное имя TFTP-сервера>"
```

Например:

```
Set-DhcpServerv4OptionValue -OptionID 66 -Value "pxe.local"
```

– для «Scope Options»:

```
Set-DhcpServerv4OptionValue -ScopeId <идентификатор скоупа> -OptionID
66 -Value "<доменное имя TFTP-сервера>"
```

Например:

```
Set-DhcpServerv4OptionValue -ScopeId <идентификатор скоупа> -OptionID
66 -Value "pxe.local"
```

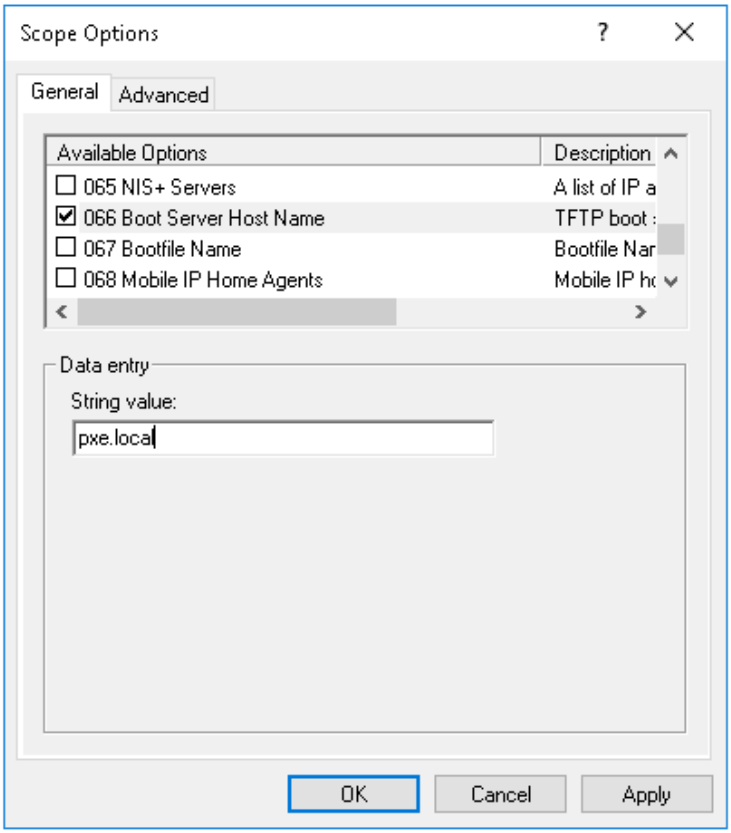


Рисунок 22

3.10.31.1.1.2 Настройка передачи имени файла-загрузчика (Option 67)

Если пользовательские ЭВМ поддерживают EFI/UEFI загрузку, для передачи имени файла-загрузчика необходимо в пункте «Configure Options» (см. Рисунок 21) выбрать опцию Bootfile Name (Option 67) и указать файл `ipxe.efi`.

Настройку передачи имени файла-загрузчика также можно осуществить с помощью «PowerShell»:

– для «Server Options» выполнить команду:

```
Set-DhcpServerv4OptionValue -OptionID 67 -Value "ipxe.efi"
```

– для «Scope Options»:

```
Set-DhcpServerv4OptionValue -ScopeId <идентификатор скоупа> -OptionID
67 -Value "ipxe.efi"
```

При использовании режима загрузки «Legacy BIOS», либо при различии архитектур, требуется настройка политик.

1) Настройка политики для «Legacy BIOS»

Для настройки политики для «Legacy BIOS» необходимо выполнить следующие действия:

– создать «Vendor Class», выбрав в контекстном меню «IPv4» пункт «Define Vendor Classes» (Рисунок 23);

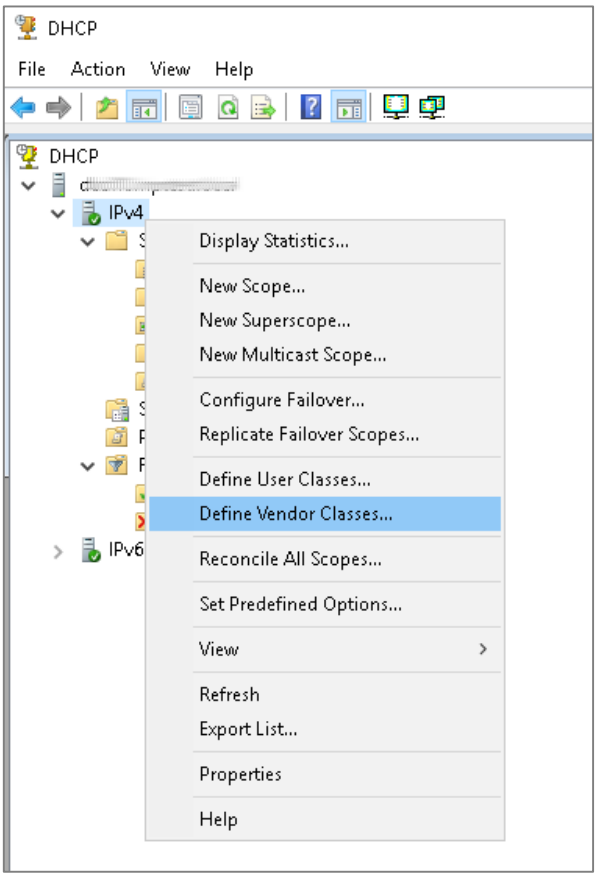


Рисунок 23

– в открывшемся окне нажать кнопку «Add» и задать тип клиентской архитектуры, который передают BIOS загрузчики при обращении к DHCP, указав в полях «Display name» и «ASCII» (Рисунок 24) значение «PXEClient:Arch:00000»;

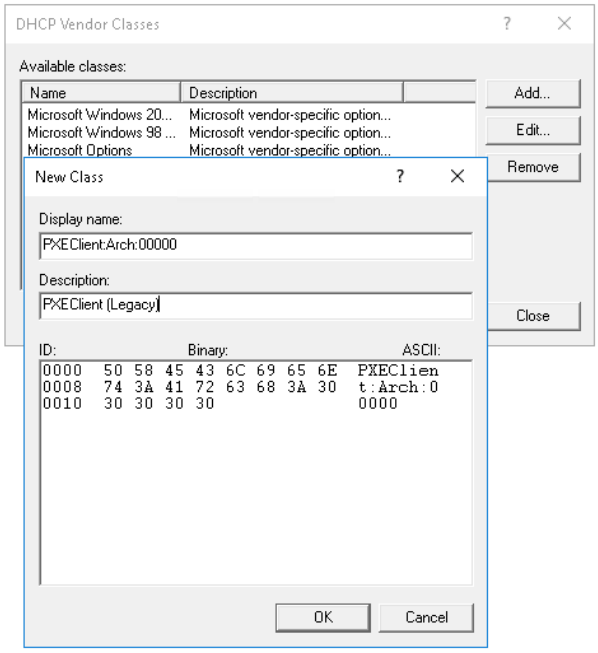


Рисунок 24

– создать политику для сопоставления класса вендора с опцией DHCP, выбрав в контекстном меню «IPv4/Policies» или «IPv4/Scope/Policies» пункт «New Policy» (Рисунок 25);

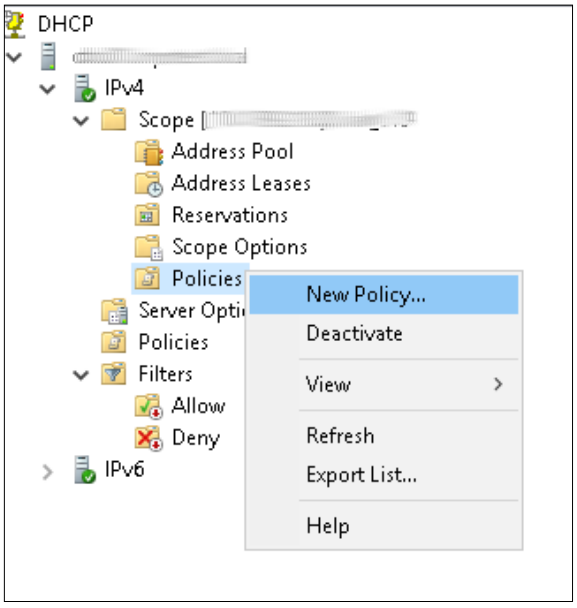


Рисунок 25

– в отобразившемся окне в поле «Policy Name» указать имя политики, например, «pxeclient-legacy» (Рисунок 26);

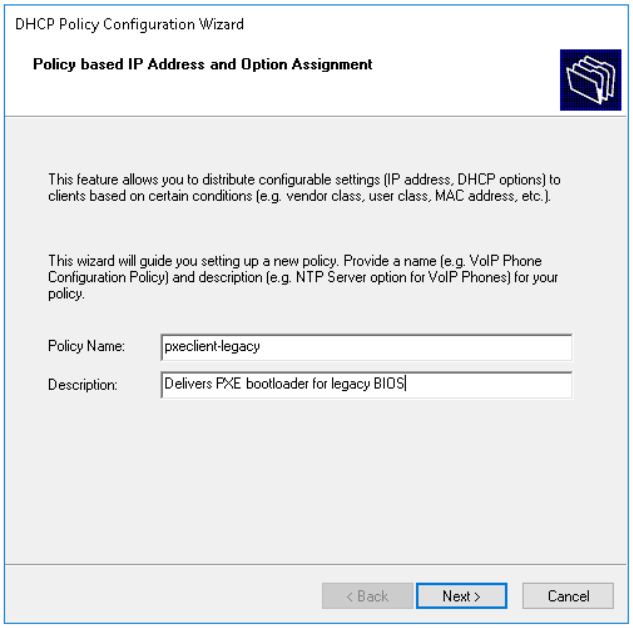


Рисунок 26

– на странице настройки условий нажать кнопку «Add». В открывшемся в поле «Criteria» выбрать значение «Vendor Class», а в поле «Operator» - «Equals» (Рисунок 27) и нажать кнопку «Ок»;

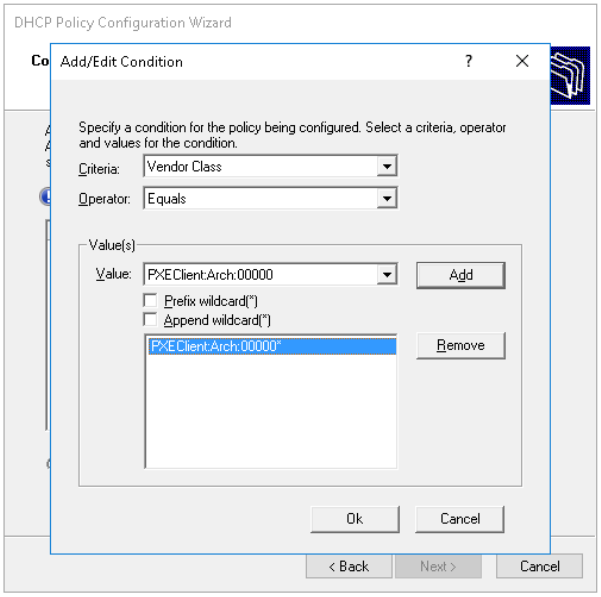


Рисунок 27

– на странице настроек политики отказаться от переопределения диапазона адресов, выбрав пункт «No»;

– на следующей странице выбрать из списка опцию 67 (Bootfile Name) и ввести значение «undionly.kpxe» (Рисунок 28);

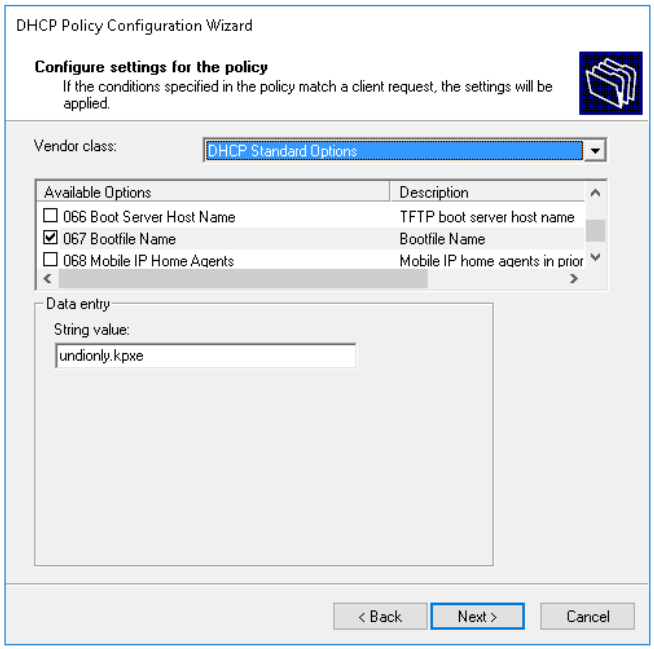


Рисунок 28

– убедиться, что все введенные значения верны, и завершить настройку политики (Рисунок 29). После выполнения указанных действий на странице «Scope Options» появится новая опция с номером 67.

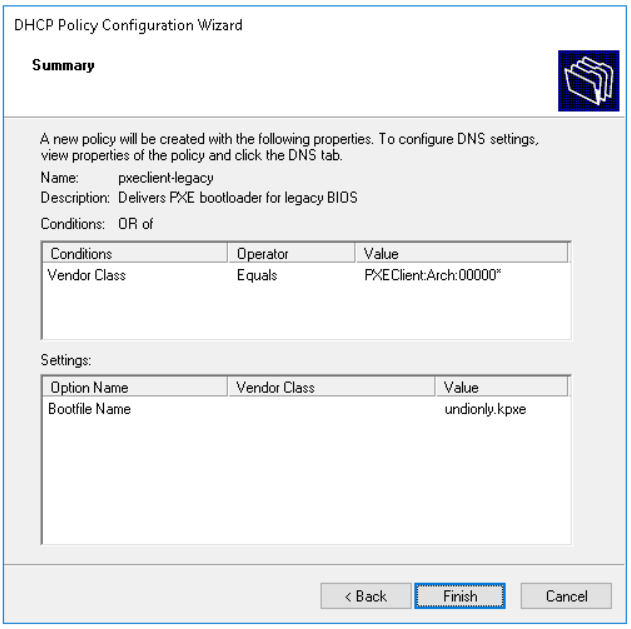


Рисунок 29

ПРИМЕЧАНИЕ. Загрузчик «undionly.kpxe» содержит скрипт, при загрузке автоматически запускающий `autoexec.ipxe` с TFTP-сервера.

Настройку политики также можно осуществить с помощью «PowerShell», выполнив команды:

```
Add-DhcpServerv4Class -Name "PXEClient:Arch:00000" -Type Vendor -
Description "PXEClient (Legacy)" -Data "PXEClient:Arch:00000"

Add-DhcpServerv4Policy -Name "pxeclient-legacy" -Condition OR -
VendorClass EQ,PXEClient:Arch:00000*

Set-DhcpServerv4OptionValue -PolicyName "pxeclient-legacy" -OptionID
67 -Value "undionly.kpxe"
```

2) Настройка политики для «EFI/UEFI»

Настройка политики для «EFI/UEFI» осуществляется аналогично настройке политики для «Legacy BIOS». В процессе настройки необходимо заменить файл-загрузчик «undionly.kpxe» на «ipxe.efi» (Рисунок 30) и идентификаторы архитектуры (Рисунок 31).

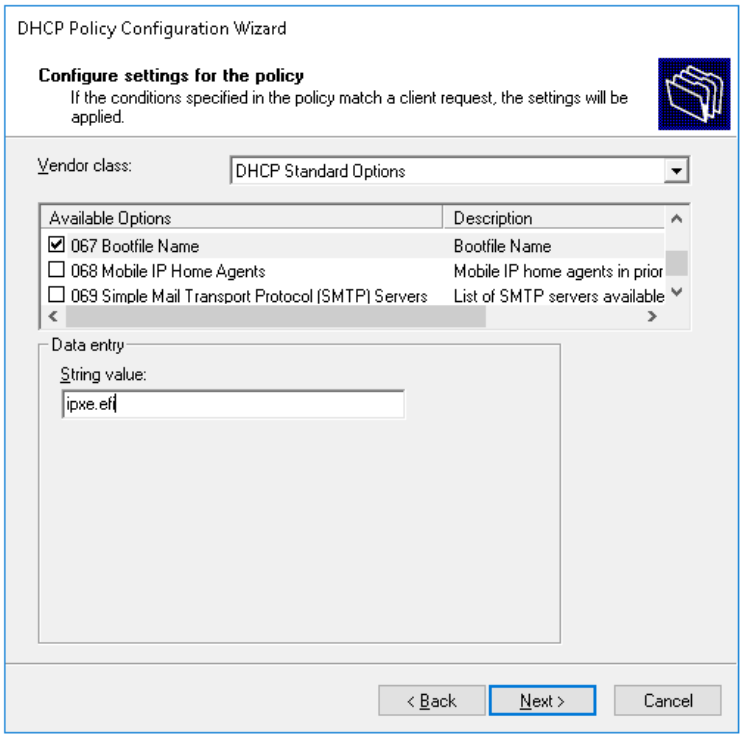


Рисунок 30

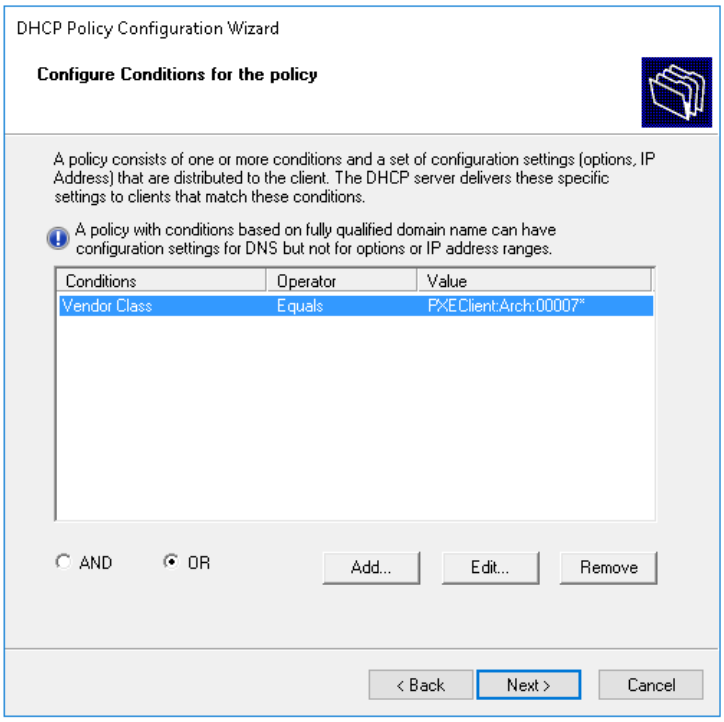


Рисунок 31

ПРИМЕЧАНИЕ. Возможно использование нескольких архитектур в рамках одной политики. Типы архитетур описаны в стандарте RFC 4578. Идентификаторы архитектур:

```
# UEFI x86
PXEClient:Arch:00002
PXEClient:Arch:00006

# UEFI x64
PXEClient:Arch:00007
PXEClient:Arch:00008
PXEClient:Arch:00009
```

Для настройки политики через «PowerShell» необходимо выполнить команды:

```
Add-DhcpServerv4Class -Name "PXEClient:Arch:00007" -Type Vendor -
Description "PXEClient (UEFI x64)" -Data "PXEClient:Arch:00007"

Add-DhcpServerv4Policy -Name "pxeclient-uefi64" -Condition OR -
VendorClass EQ,PXEClient:Arch:00007*

Set-DhcpServerv4OptionValue -PolicyName "pxeclient-uefi64" -OptionID
67 -Value "ipxe.efi"
```

3.10.31.1.2 Настройка DHCP-сервера с использованием программы Dnsmasq

В качестве DHCP-сервера может использоваться программа `dnsmasq`. Пример конфигурации:

```
dhcp-match=set:efi-x86_64,option:client-arch,7  # определение
архитектуры клиента(EFI)

dhcp-match=set:efi-x86_64,option:client-arch,9  # определение
архитектуры клиента(EFI)

dhcp-match=set:bios,option:client-arch,0        # определение
архитектуры клиента(BIOS)

dhcp-match=set:iPXE,175                         # определение того,
что пакет пришел от загрузчика iPXE

dhcp-match=set:iPXE,77

# назначение загрузчиков в зависимости от архитектуры клиента
dhcp-boot=tag:efi-x86_64,"ipxe.efi"
dhcp-boot=tag:bios,"undionly.kpxe"
dhcp-boot=tag:iPXE,"autoexec.ipxe"
```

3.10.31.1.3 Настройка DHCP-сервера с использованием программы DHCPD

В качестве DHCP-сервера может использоваться программа DHCPD. Пример конфигурации для `isc-dhcp-server`:

```
# определение новой опции - архитектура клиента (код 93 по RFC4578)
option client-architecture code 93 = unsigned integer 16;

# если клиент представился как "iPXE" - отдать загрузочное меню
if exists user-class and option user-class = "iPXE" {
  filename "autoexec.ipxe";

# если архитектура клиента - 0 (STandard PC BIOS) - отдать загрузчик
для legacy
} elsif option client-architecture = 00:00 {
  filename "undionly.kpxe";

# во всех прочих случаях считать, что клиент - EFI/UEFI
} else {
  filename "ipxe.efi";
}
```

3.10.31.2. Установка и настройка PXE-сервера

Для установки PXE-сервера необходимо выполнить следующие действия:

– отредактировать конфигурационный файл `config/vars/_vars.yml`, указав в переменной `ipxe_server_name` адрес или доменное имя, по которому ЭВМ пользователей будут обращаться к PXE-серверу, например:

```
ipxe_server_name: ocs-pxe-server.domain
```

– отредактировать конфигурационный файл `config/vars/_vars.yml`, добавив переменные (без вложений в другие словари) с паролями от создаваемых учетных записей при установке ОС. Например:

```
pxe_guest_system_grub_password: "12345678" # Пароль для GRUB
pxe_guest_system_root_password: "12345678" # Пароль для root-
пользователя
pxe_guest_system_sysadmin_password: "87654321" # Пароль для первичного
sysadmin-пользователя
pxe_guest_system_sysadmin_password_hash:
"$6$OEyU9JwGeh.P$lRyNBvmb235xKqdtzhVPeF.q8iZN37IAKZcKM4b5MWtS9KiIuAyX8
o4Id94bsG7smD5SVUe3ZR2NLLW22Trv4." # Хэш-пароля sysadmin-пользователя.
Используется, если требуется записывать пароль не в открытом виде
```

ПРИМЕЧАНИЕ. Хэш-пароля для переменной `pxe_guest_system_sysadmin_password_hash` можно получить, используя следующую команду:

```
mkpasswd -m sha-512 <пароль> <соль>
```

ВАЖНО! Добавление указанных переменных определяется их наличием в preseed-файле. Если переменные в preseed-файле не указаны, их не требуется добавлять и заполнять.

– в переменной `pxe_os_list` указать список конфигураций ОС, которые будут доступны в загрузочном меню iPXE, например:

```
pxe_os_list:
  # Имя конфигурации
  - name: alt_p10
    iso:
      # Имя файла ISO
      file: alt-workstation-10.1-x86_64.iso
      # Хеш-сумма для проверки целостности образа (опционально).
```

```
      # При отсутсвии хэша проверка не проводится и файл по умолчанию
считается корректным.
      sha1: d6d2e7cf551ab2045c07d19a95d2efbcce25f5ee
      # URL для загрузки ISO образа по HTTP (опционально).
      # Если образ присутствует на PXE сервере и проверка целостности
завершена успешно (или не проводилась), повторная его загрузка с
управляющей машины на PXE не запустится.
      # Если образ присутствует на управляющем хосте и проверка
целостности завершена успешно (или не проводилась), повторная его
загрузка по указанному URL не запустится.
      # Если URL не указан, поиск образа осуществляется только
локально в каталоге pxe/<имя конфигурации>/iso/.
      #url: https://<web_storage_domain>/alt-p10/alt-workstation-10.1-
x86_64.iso

    # Список файлов для конфигурации установщика.
    # Элементы списка могут быть указаны в виде URL или имён файлов. В
первом случае файл будет скачан, во втором - взят из локального
каталога.
    # Если файл имеет расширение `.j2`, т.е. является шаблоном Jinja2,
все переменные будут предварительно раскрыты и создан его итоговая
копия без расширения `.j2`.
    # ВАЖНО: при наличии файла Jinja каждый запуск будет приводить к
замене его итоговой копии. Для внесения право нужно править файл-
шаблон.
    assets:
      # Преконфигурационные файлы для загрузки в EFI/UEFI режиме.
      metadata_efi:
        #- https://<web_storage_domain>/alt-p10/efi/autoinstall.scm.j2
        #- https://<web_storage_domain>/alt-p10/efi/vm-profile.scm
        - autoinstall.scm.j2
        - vm-profile.scm
      # Преконфигурационные файлы для загрузки в Legacy BIOS режиме.
      metadata_pcbios:
        #- https://<web_storage_domain>/alt-
p10/pcbios/autoinstall.scm.j2
        #- https://<web_storage_domain>/alt-p10/pcbios/vm-profile.scm
        - autoinstall.scm.j2
        - vm-profile.scm
      # Список скриптов для пост-инсталляции
      postinstall:
        #- https://<web_storage_domain>/alt-
p10/alt_p10_workstation.sh.j2
        - alt_p10_workstation.sh.j2

    # Список необходимых файлов, доступных в ISO образе (опционально)
    iso_assets:
      # Пути задаются относительно корня файловой системы внутри ISO
      metadata_efi:
        - Metadata/pkg-groups.tar
      metadata_pcbios:
        - Metadata/pkg-groups.tar

  # Другая конфигурация
```

```
  - name: ...
    ...
```

– создать в корневой директории дистрибутива следующую структуру каталогов в соответствии со списком `pxe_os_list`:

```
pxe
├── <имя_конфигурации>
│   ├── iso
│   │   └── <image>.iso
│   ├── metadata
│   │   ├── efi
│   │   │   └── <autoinstall>.[j2|scm|cfg|...]
│   │   └── pcbios
│   │       └── <autoinstall>.[j2|scm|cfg|...]
│   └── postinstall
│       └── <postinstall script>
├── ...
│   ├── iso
│   ├── metadata
│   └── postinstall
...
```

– для каждой ОС в каталогах `pxe/<имя конфигурации>` создать подпапки:

- «iso» с установочным iso-образом соответствующей ОС;
- «metadata» с файлом или файлами автоответов инсталлятора на основе документации ОС;
- «efi» - подпапка каталога «metadata» с файлами загрузчика для EFI/UEFI;
- «pcbios» - подпапка каталога «metadata» с файлами загрузчика для BIOS;
- «postinstall» - shell-скрипты, которые выполнятся при окончании инсталляции (опционально);

ПРИМЕЧАНИЕ. Для ОС Альт Linux можно найти примеры предзаполненных файлов (`autoinstall.scm.j2`, `vm-profile.scm`, `alt_p10_workstation.sh.j2`) в директории `install-ac-*/samples/pxe/alt_p10/metadata`. Файлы необходимо отредактировать с учетом собственных потребностей в соответствии с документацией производителя ОС и убрать расширение «.j2». Описание файлов и

настроек представлено на сайте разработчика ОС Альт: https://docs.altlinux.org/ru-RU/alt-workstation/10.1/html/alt-workstation/install-distro--autoinstall--chapter.html.

– установить на PXE-сервер необходимые пакеты с помощью команды:

```
ansible-playbook play-managed-node-prerequisites.yml -i
inventories/hosts.yml -vv --diff -l pxe -u "<имя пользователя>"
```

Например:

```
ansible-playbook play-managed-node-prerequisites.yml -i
inventories/hosts.yml -vv --diff -l pxe -u omp
```

– установить PXE-сервер с помощью команды:

```
ANSIBLE_USER="<имя пользователя>" ./deploy-infra.sh -c nginx,pxe -l
pxe
```

Например:

```
ANSIBLE_USER=omp ./deploy-infra.sh -c nginx,pxe -l pxe
```

3.10.31.3. Установка ОС по сети на ЭВМ пользователей

Для установки ОС на ЭВМ пользователей необходимо:

– запустить ЭВМ пользователя, выбрав приоритетной загрузку по сети;

– в открывшемся меню выбрать установку требуемой ОС;

– после завершения установки подключить ЭВМ пользователя к ППО с помощью активационного файла или строки, полученных при инициации процесса активации устройства в ПУ (в соответствии с документом «Руководство пользователя. Часть 3. Подсистема Платформа управления»), выполнив команду:

```
/usr/bin/omp-uem-client activate --json='<активационная строка>'
# или
/usr/bin/omp-uem-client activate --file=<файл активации>
```

3.10.32. Настройка подключения flatpak репозиториев

Flatpak репозитории используются для установки приложений на управляемые ЭВМ.

ЭВМ могут взаимодействовать с flatpak репозиторием как напрямую, так и через контент-сервер ППО. Взаимодействие с flatpak репозиторием через контент-сервер требуется в тех случаях, когда необходим аутентифицированный доступ к репозиторию.

Способ взаимодействия определяет порядок настройки подключения ППО к flatpak репозиторию. Подключение flatpak репозитория осуществляется с помощью политики «Конфигурация репозиториев/Подключение flatpak репозиториев». Порядок работы с политиками приведен в документе «Руководство пользователя. Часть 3. Подсистема Платформа управления».

В случае, если взаимодействие с flatpak репозиторием осуществляется через контент-сервер, то необходимо дополнительно выполнить следующие дейтвия:

1) В секции `config.publicUris.pkgrepo.linuxRepoAddresses` конфигурационного файла `config/config.yml.j2` задать его адрес, имя и тип, например:

```
pkgrepo:
# Linux-репозитории, содержимое которых предоставляется через контент
серверы ППО
    linuxRepoAddresses:
        - name: "test-flatpak" # Уникальное имя репозитория
(используется только символы, разрешенные для URL без кодировки)
          type: "flatpak" # Тип репозитория в зависимости от пакетов:
rpm, deb, flatpak
          address: http://ocs-flatpak.local # Адрес репозитория
```

ВНИМАНИЕ! Имя репозитория может содержать только цифры, заглавные и строчные буквы (кириллица не допускается).

2) Переустановить конфигурационные файлы ПООС и CDN с помощью команды:

```
ANSIBLE_USER="<имя пользователя>" ./deploy-ac.sh --action config --
subsystems pkgrepo,cdn
```

3) Подготовить конфигурационный файл `<имя файла>.flatpakrepo` с настройками подключения к репозиторию через контент-сервер, содержащий следующие параметры:

– `Title` – название репозитория;

– `Url` – путь к репозиторию в формате: `<фактический адрес контент сервера>/pkgrepo/linux/flatpack/<название репозитория, согласно параметру config.publicUris.linuxRepoAddresses.name конфигурационного файла config/config.yml.j2>/<путь к репозиторию>;`

– `Comment` – кртакий комментарий к репозиторию (опциональный параметр);

– `Description` – подробное описание репозитория (опциональный параметр);

– `GPGKey` – открытый ключ репозитория, который используется для проверки подлинности пакетов, загружаемых из репозиториев (опциональный параметр).

Например:

```
[Flatpak Repo]
Title=Protected repo for testing
Url=https://example.ac.ru/pkgrepo/linux/flatpak/test-flatpak-on-
premise/repo/
Comment=Some comment
Description=Some descriptinon
GPGKey=xsBNBGjs4z4BCACh0FjyZHQhep2YXSrMx/Tv+xg263d2HK0w/WPhkkGic0lb
bxDMwNw7Grrl6J2E8/D2h4qHdPGzAcTTTk5ZPlGESlmIA4LnBbjwBVSBog80
QKGz7lE3f69ieN3o1Tc3MfVSi774Tu6sqayR0ZMWHki++8QH8GykE+VW5W9Y
SBWUBWPyJIrFhohyx0RaqVG3AWDwUOCWrXzc8unznv1sPzg+LEnf0Fwu62tZ
wkv4B92kyUD3F2Adma2kR3NEwZND03ub0v1IRtCKh0nK6/foe3gFPtjg1w1G
n4EHdRftjJsh5nPnAeYF3OpJgRfCbDb+4P0n+x97fvR9NwuMLasq5BcBABEB
AAHNDjEyM0BzZGZzYS5yc2RmwsCKBBABCAA+BYJo7OM+BAsJBwgJkFmR2sNB
OfM+AxUICgQWAAIBAhkBApsDAh4BFiEEn5uND50Wm/yNBjfIWZHaw0E58z4A
AC5jB/43FPSShXhqHkF+fD0o1mf8gRQJod3//DbNcKckAqZUEWS3+Ny7d8i/
+s3YaQKh4TwPYkE/VHyz19qwL6fWDtJ6/x0DCWHhypEV+XHTPtqUa+A1+MXe
llsxy6rzfX7IIIpZyvt2jz6K3nRRkOIUcBD9/jg+F+ag0rBqR7gmLJ8gQO0j
KSB1ra1q3/LlZEquOKdUNMkGc0zwr1I4xHidQsFPtxJW+UknF5Mcn3cYnfUw
0P9oaNx8WYV8jg+x0ZPbSk7LKWucvH/ggcoG6HFWfThubCdhWh2wVFVQKTbD
sTR3irDL/ljNva1l9DD3E7zqoXG7AYe6/047s9dBoVmnhJByzsBNBGjs4z4B
CACopdRsfdIl+L5+iQJAj/Hg7TarpC6vsvSx+UQiLOJcNyQ/9NMnOhTtJqgl
FS4/B8FMShKGP5xWQUwmAO+RJmkNcCdvfXwKR9YB9A3VfEw3taXV01et2gsd
M5ThMcxxqHjK501AxxgcWDMVjbdvE/j92Vqb/dOJU42FExE/Vq9HfXi50yHg
CUsqkHUC5RcYn96E2b4bTdUA/l8AubIoO85keiJgBXUuzmBgIrK0uZ8JfWth
uX/cxiK37R8Lhwej49f45EgP5HX4IzZL4voqvhEIMibD4TbF2P/lJa+HwfR0
Frp7IYmwLtcrx4EJwn7RN9wWI4v2WacMOHP04mc21puvABEBAAHCwHYEGAEI
```

```
ACoFgmjs4z4JkFmR2sNBOfM+ApsMFiEEn5uND50Wm/yNBjfIWZHaw0E58z4A
AIsmB/9a+A9TJdG+2gRJ6Zcf009Cx+DnPmf3KmB0dHQ6tXkeU97u93fJI9zY
96TbRYsdKhSoO7wAo7aN/iCBCaqqpqljbKhcOx7mvRyFu8n8iW/Z2FMeCSNb
uokKYfPjBji20mSivitj+cPVJya2GJB40/CsHJCbVBzAJg+c39TXhyzBaBB7
qAB0nAQ5ZTChh5Ji73x6SR4U1VK4KgzI14Cs1uizGtYk9QR0kGiielLPtHu9
PsQknOqEtKjQ8xBRXlsIkTSSe4MzkUMW+iS7EQqS1ieWIxE5eZ5dKKn4ukZG
8A5+j/Dwe7i2YzEEeI44UBs53z45HrBjau7ss+pux8HF2xFN
=ifPZ
```

4) Разместить конфигурационный файл `<имя файла>.flatpakrepo` в flatpack репозитории. В дальнейшем адрес к данному файлу необходимо указать в политике «Конфигурация репозиториев/Подключение flatpak репозиториев». Порядок работы с политиками приведен в документе «Руководство пользователя. Часть 3. Подсистема Платформа управления».

3.10.33. Настройка интеграции с S3 хранилищем

ППО поддерживает возможность интеграции с S3-совместимым хранилищем для хранения файлов в качестве альтернативы файловой системе.

ПРИМЕЧАНИЕ. Для корректной настройки интеграции должно быть развернуто S3 хранилище и создан отдельный бакет для данных ППО.

ВНИМАНИЕ! При интеграции ППО с S3 хранилищем не будет осуществляться загрузка данных в файловое хранилище ПООС и синхронизация с git-репозиторием для получения папок и файлов. Для реализации функционала ПООС требуется использовать хранилище другого типа - локальный диск, либо NFS.

Для настройки интеграции необходимо:

1) В блоке `fileStorageSetting` конфигурационного файла `config/config.yml.j2` задать следующие параметры:

- `type` – тип хранилища файлов, должен иметь значение `objectStorage`;
- `useSSL` – использование шифрования запросов;
- `address` – адрес S3-хранилища;
- `bucketName` – название бакета.

Пример блока «fileStorageSetting»:

```
#
# Настройки хранения файлов
#
fileStorageSettings:
  type: "objectStorage" # тип хранилища файлов
                        #   - fileSystem - хранение на файловой
системе
                        #   - objectStorage - хранение в S3-
совместимом хранилище
  objectStorage:
    useSSL: true                              # Использование
шифрования запросов
    address: "ocs-s3.local:8080"              # Адрес объектного
хранилища
    bucketName: "example-backet-name"         # Название бакета с
которым будет работать АЦ
```

2) В блоке `fileStorageSetting` конфигурационного файла `config/secret.yml` задать идентификатор ключа и ключ доступа к S3-хранилищу, изменив параметры `accessKey` и `secretKey`, например:

```
#
# Настройки хранения файлов
#
fileStorageSettings:
  objectStorage:
    accessKey: "KeyID"
    secretKey: "Password"
```

3.10.34. Порядок получения метрик ППО

ППО поддерживает возможность сбора метрик на шлюзах доступа в формате Prometheus. Сбор метрик осуществляется коллектором телеметрии (OpenTelemetry Collector), который формирует общий набор метрик и публикует его на порту 8081 сервера приложений, эндпоинт `/metrics`.

Для получения всех метрик необходимо выполнить команду:

```
curl http://<сервер приложений>:8081/metrics
```

Например:

```
curl http://ocs-app.local:8081/metrics
```

Для получения метрик определенного шлюза необходимо выполнить команду:

```
curl <gateway>/admin/metrics
```

Например:

```
curl ocs-auth-public-api-gw.local/admin/metrics
```

Полученные данные могут быть импортированы в Grafana в соответствии с разделом 11.

ПРИМЕЧАНИЕ. Пример файла отчета для импорта в Grafana находится в каталоге `samples/dashboards/krakend-telemetry-dashboard.json`.

В среде Kubernetes сбор метрик выполняется коллектором телеметрии (OpenTelemetry Collector). Он обнаруживает Pod'ы шлюзов в кластере, получает от них метрики, обрабатывает и публикует единым набором.

Общая схема работы:

```
Gateway Pod [ <pod>:8081/admin/metrics ] – HTTP --> OpenTelemetry
Collector [ telemetry:8081/metrics ] – HTTP --> Prometheus
```

Метрики шлюзов

Каждый шлюз предоставляет метрики по порту 8081 на своих Pod'ах, эндпоинт `/admin/metrics`. Для прямого обращения к метрикам определенного шлюза используется адрес http://<IP-адрес-Pod>:8081/admin/metrics.

Коллектор обнаруживает Pod'ы шлюзов автоматически с использованием `Kubernetes Service Discovery` с типом `pod` и опрашивает каждый с интервалом в 60 секунд. После обработки `OpenTelemetry Collector` публикует собранные данные в формате `Prometheus` по адресу `0.0.0.0:8081/metrics`. Данный эндпоинт доступен через сервис `telemetry` с типом `ClusterIP`.

Доступ к метрикам внутри Kubernetes

Для получения собранных метрик изнутри кластера Kubernetes необходимо обратиться по полному FQDN адресу: http://telemetry.<namespace>.svc.cluster.local:8081/metrics.

При нахождении внутри того же неймспейса (`namespace`) доступно получение по короткому имени: http://telemetry:8081/metrics.

Для проверки метрик вне кластера необходимо предварительно выполнить проброс порта с помощью утилиты kubectl.

Шаблон команд:

```
 kubectl -n <namespace> port-forward svc/telemetry 8081:8081 curl http://localhost:8081/metrics
```

Например:

```
kubectl -n k8s-dev port-forward svc/telemetry 8081:8081 curl http://localhost:8081/metrics
```

Подключение Prometheus

Для сбора метрик необходимо настроить Prometheus на получение данных с эндпоинта `OpenTelemetry Collector`.

Например:

```
http://telemetry.k8s-dev.svc.cluster.local:8081/metrics
```

ПРИМЕЧАНИЕ. Таким образом, Prometheus получает единый поток метрик от Collector, а не обращается непосредственно к каждому отдельному шлюзу.

3.10.35. Настройка удаленного пробуждения компьютера в КПСД

Для настройки удаленного пробуждения устройств (для выполнения политики с правилом «Приложения/Управление приложениями» или «Система/Обновление ОС) требуется задать в блоке `config` конфигурационного файла `/var/ocs/config/subsystems/emm/applications/ocs-emm-state-manager-api/ocs-emm-state-manager-api.yml` следующие параметры:

```
wakeOnLanSettings:
  enabled:           # Включение/выключение удаленного пробуждения.
Состояние по умолчанию: выключено.
  workerInterval:      # Интервал времени между запусками воркера,
отвечающего за выбор будящих устройств.
  maxTargetDevices:   # Максимальное количество устройств, которые в
каждом сегменте будут назначены каждому пробуждающему устройству.
```

Например:

```
wakeOnLanSettings:
  enabled: true           # Включение/выключение удаленного
пробуждения. Состояние по умолчанию: выключено.
  workerInterval: 10s      # Интервал времени между запусками воркера,
отвечающего за выбор будящих устройств.
  maxTargetDevices: 253   # Максимальное количество устройств, которые
в каждом сегменте будут назначены каждому пробуждающему устройству.
```

Перезапустить сервис `ocs-emm-state-manager-api` с помощью команды:

```
systemctl restart ocs-emm-state-manager-api*
```

3.10.36. Настройка проверки срока жизни управляемых переменных

Управляемая переменная представляет собой ключ и значение, которые хранятся в защищенном хранилище сервера ППО с последующей передачей через политики на устройства. Управляемые переменные используются при выполнении скриптов, настройках конфигураций приложений и LBS-сервисов на устройствах. Также у управляемых переменных может задаваться срок жизни, по истечению которого они блокируются и устройство перестает их принимать. Для того, чтобы отслеживать срок жизни управляемых переменных в ППО есть возможность задавать интервалы массовой проверки.

Для настройки проверки срока жизни управляемых переменных требуется указать количество потоков для обработки запроса и интервал времени между запусками массовой проверки срока жизни управляемых переменных. Для этого в конфигурационном файле `config/subsystems/emm/applications/ocs-emm-policies-api/ocs-emm-policies-api.yml` необходимо задать параметры

config.managedVariables.workersCount и config.managedVariables.workerSleepInterval соответственно, например:

```
config:
  managedVariables:
    workersCount: 1                  # Количество потоков,
обрабатывающих запрос.
    workerSleepInterval: '1m'        # Интервал запуска проверки срока
жизни для управляемых переменных. Значение в минутах. Минимальное
значение - 1m, максимальное значение - 60m.
```

Перезапустить сервис ocs-emm-policies-api с помощью команды:

```
systemctl restart ocs-emm-policies-api*
```

3.10.37. Настройка интеграции с внешней системой Platform V IAM SE

Данная интеграция позволяет через веб-интерфейс ППО осуществлять аутентификацию и авторизацию пользователей, которые управляются внешней IAM-системой Platform V IAM SE.

ВНИМАНИЕ! Интеграция успешно протестирована на версиях Platform V IAM SE 3.2.0 и IAM Proxy 4.6.0. Совместимость с другими версиями также возможна.

При включенной интеграции:

– ППО делегирует во внешнюю IAM-систему Platform V IAM SE функции аутентификации учетной записи пользователя ППО (валидацию учетных данных, срока действия пароля, неудачных попыток входа), учета количество параллельно запущенных сессий привилегированных учетных записей и проверку количество дней неактивности учетной записи с ее дальнейшей блокировкой;

– ППО регистрирует учетную запись по логину и списку ролей, полученных из внешней IAM-системы при первичной аутентификации;

– ППО ограничивает возможность аутентификации локальных учетных записей, управления локальными учетными записями (создание, обновление, удаление и блокировку доступа). Управление учетными записями осуществляется централизовано через внешнюю IAM-систему. Исключениями являются случаи, когда происходит первая установка ППО с последующим автоматическим созданием

учетной запись admin@omp.ru и когда вручную создается техническая учетная запись (ТУЗ);

– ППО обеспечивает аутентификацию учетной записи по полученному IAM-токену для создания внутренней пользовательской сессии и авторизацию пользователей при их обращении через IAM Proxy в аутентифицированном виде. Авторизация пользователя выполняется для каждого запроса в соответствии с полученными данными (логин и список ролей) в IAM-токене.

При выключенной интеграции ППО продолжает свою работу без изменений. Доступна идентификация, аутентификация и авторизация локальных учетных записей, а также возможность управления (создание, обновление, удаление и блокировка доступа) ими.

Настройка интеграции с внешней IAM-системой устанавливается в конфигурационном файле config/enviroments/dropapp/config.yml в секции IAMConfig следующими параметрами:

```
IAMConfig:
  enabled: true          # Включение/Выключение интеграции с IAM
  audience: ""           # Адресат IAM-токена
  issuer: ""             # Эмитент IAM-токена
  logoutUri: ""          # Адрес для выполнения выхода из внешней IAM
  jwksUri: ""            # URL для получения списка публичных ключей
для проверки подписи IAM-токена
  jwksUriTimeout: ""     # Таймаут для получения списка публичных
ключей
  jwksCacheDuration: "" # Продолжительность кэширования списка
публичных ключей
  tokenHeaderName: ""    # Название заголовка, содержащего IAM-токен
```

Например:

```
IAMConfig:
  enabled: true
  audience: "AuroraCenter-dropapp"
  issuer: "<iam-server-url>"
  logoutUri: "<iam-server-url>/logout?revoke=true&logout_idp=true"
  jwksUri: "<iam-server-url>/certs"
  jwksUriTimeout: "60s"
  jwksCacheDuration: "15m"
  tokenHeaderName: "IAMTokenHeader"
```

После внесения изменений необходимо переустановить конфигурационные файлы ПБ с помощью команды:

```
ANSIBLE_USER="<имя пользователя>" ./deploy-ac.sh --action config --
subsystems auth
```

ППО выполняет аутентификацию и авторизацию пользователя на основе данных из IAM-токена в соответствии с полями `subject` и `roles`. При получении запроса, в котором IAM-токен содержит измененный список ролей, ППО обновит информацию о ролях пользователя и выполнит последующую авторизацию в соответствии с актуальными ролями.

ПРИМЕЧАНИЕ. Сторона, использующая ППО и осуществляющая интеграцию, должна внести роли во внешнюю IAM-систему и назначить учетным записям пользователей. При назначении ролей учетной записи пользователя не допускаются пересечения между следующими ролями:

– Администратор Платформы управления (EMM_ADMIN);

– Администратор устройств Платформы управления (EMM_DEVICE_ADMIN);

– Специалист технической поддержки Платформы управления (EMM_SUPPORT);

– Специалист технического обслуживания устройств Платформы управления (EMM_DEVICE_SUPPORT).

При получении запроса с IAM-токеном, в котором пересекаются эти роли, ППО ответит ошибкой `IDP_ROLE_CONFLICT`.

В таблице (Таблица 20) приведены роли доступные для назначения.

Таблица 20

Роль в веб-интерфейсе ППО	Роль в справочнике ППО	Перечень ролей недоступных для пересечения
Разработчик	APS_DEVELOPER	-
Администратор Аврора Маркет	APS_ADMIN	-

Роль в веб-интерфейсе ППО	Роль в справочнике ППО	Перечень ролей недоступных для пересечения
Редактор приложений	APS_APPLICATIONS_EDITOR	-
Пользователь Аврора Маркета	APS_DEVICE_USER	-
Администратор учетных записей	AUTH_ADMIN	-
Оператор аудита	AUTH_AUDITOR	-
Администратор тенантов	MT_ADMIN	-
Администратор Сервиса уведомлений	PNS_ADMIN	-
Администратор Платформы управления	EMM_ADMIN	EMM_DEVICE_ADMIN EMM_SUPPORT EMM_DEVICE_SUPPORT
Администратор устройств Платформы управления	EMM_DEVICE_ADMIN	EMM_ADMIN EMM_SUPPORT EMM_DEVICE_SUPPORT
Специалист технической поддержки Платформы управления	EMM_SUPPORT	EMM_ADMIN EMM_DEVICE_ADMIN EMM_DEVICE_SUPPORT
Специалист технического обслуживания устройств Платформы управления	EMM_DEVICE_SUPPORT	EMM_ADMIN EMM_DEVICE_ADMIN EMM_SUPPORT

Более подробное описание каждой роли приведено в соответствующем приложении документа «Руководство Пользователя. Часть 1. Подсистема безопасности».

ПРИМЕЧАНИЕ. Роль «Администратор учетных записей» предоставляет полномочия для управления учетными записями пользователей в ППО. Данную роль не рекомендуется назначать учетным записям пользователей для выполнения стандартных задач. При этом эту роль необходимо назначить при создании ТУЗ.

Рекомендуемые шаги:

1) Администратор внешней IAM-системы выдает роль «Администратор учетных записей» для учетной записи уполномоченного сотрудника;

2) Уполномоченный сотрудник переходит в веб-интерфейс ППО и создает необходимые ТУЗ;

3) При завершении процесса создания ТУЗ, администратор внешней IAM-системы отзывает роль «Администратор учетных записей» у учетной записи уполномоченного сотрудника.

ВНИМАНИЕ! При выходе пользователя из ППО сначала завершается его сессия в ППО, затем автоматически направляется запрос на завершение сессии во внешней IAM-системе. После того, как внешняя IAM-система успешно завершит пользовательскую сессию, он не сможет получить доступ к функционалу ППО. Для дальнейшего использования ППО пользователю необходимо будет повторно пройти процесс аутентификации во внешней IAM-системе.

3.10.38. Настройка проверки для очистки файлов билдов и зависимостей из хранилища

Для настройки интервала проверки с дальнейшей очисткой файлов билдов и зависимостей, не привязанных ни к одному из релизов приложений, требуется задать в секции `config` конфигурационного файла `/var/ocs/config/subsystems/appstore/applications/ocs-appstore-applications-api/ocs-appstore-applications-api.yml` параметр `fileRemoveInterval`.

Например:

```
config:
  #-----------------------------------------------------------------
  # Настройки хранения файлов
  #-----------------------------------------------------------------
  fileRemoveInterval: 24h # Интервал запуска проверки для очистки файлов билдов и зависимостей не привязанных ни к одному из релизов приложений. Минимальное значение 1h.
```

После изменений в конфигурационном файле необходимо перезапустить сервис, чтобы внесенные изменения вступили в силу. Для этого необходимо воспользоваться следующей командой:

```
systemctl restart ocs-appstore-applications-api*
```

3.10.39. Настройки очистки данных при удалении устройств

Для настройки очистки данных при удалении устройств требуется задать в секции `cleanup` конфигурационного файла `/var/ocs/config/subsystems/emm/config.yml` параметры:

– `pollInterval`. Интервал опроса для очистки. Возможные единицы измерения: "ns", "us", "ms", "s", "m", "h". Минимальное значение от 1 минуты, а максимальное значение не должно превышать 24 часа;

– `delay`. Задержка перед удалением данных. Она должна быть больше, чем время жизни токена для устройства (по умолчанию 12 часов). Возможные единицы измерения: "ns", "us", "ms", "s", "m", "h". Минимальное значение строго больше 12 часов, а максимальное значение не должно превышать 4320 часов.

Например:

```
#-----------------------------------------------------
# Настройки очистки данных при удалении устройств
#-----------------------------------------------------
cleanup:
  pollInterval: "5m"  # Интервал опроса для очистки
  delay: "24h"        # Задержка перед удалением данных. Должна быть
больше, чем время жизни токена для устройства (по умолчанию 12 часов)
```

После внесения изменений необходимо переустановить конфигурационные файлы ПУ с помощью команды:

```
ANSIBLE_USER="<имя пользователя>" ./deploy-ac.sh --action config --
subsystems emm
```

3.11. Проверка корректности установки и функционирования ППО

3.11.1. Общие сведения

В целях проверки корректности установки и функционирования ППО, а также среды функционирования ППО, в состав сценариев установки включена утилита для формирования диагностического отчета.

Для формирования диагностического отчета необходимо перейти в каталог со сценариями установки `install-<версия ППО>/install-ac-mt/` и выполнить команду:

```
ansible-playbook play-diagnostic-report.yml -i inventories/hosts.yml -vv --user <имя пользователя>
```

либо команду, если требуется задать окружение:

```
ansible-playbook play-diagnostic-report.yml -i inventories/<название окружения>/hosts.yml -vv --user <имя пользователя> --extra-vars "stage=<название окружения>"
```

В результате в каталоге `report` будет сформирован файл в `report.html`.

Диагностический отчет формируется в виде файла в формате `.html` и содержит следующие разделы:

– общая информация о статусе сервисов ППО;

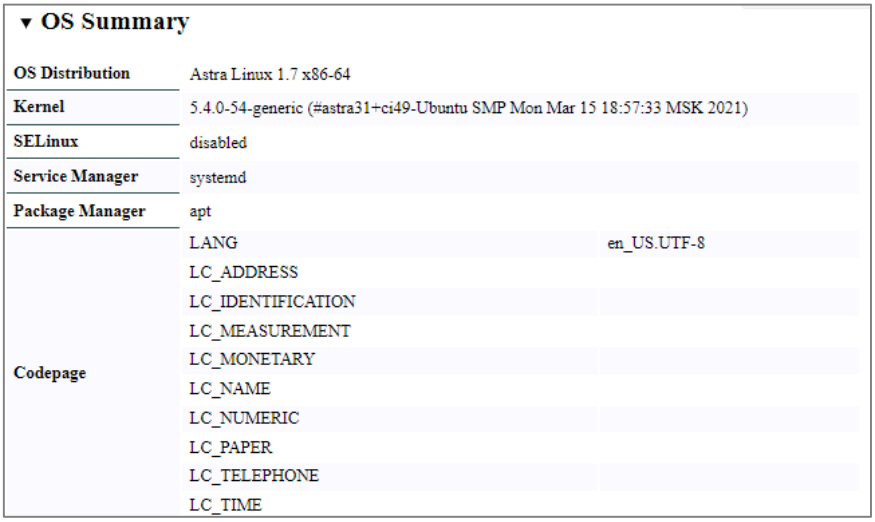
– общая информация о статусе компонентов среды функционирования ППО и инфраструктурных компонентов ППО;

– разделы, содержащие детальную информацию об отдельных сервисах ППО, компонентах среды функционирования ППО и инфраструктурных компонентах ППО.

3.11.2. Описание параметров диагностического отчета

3.11.2.1. Раздел «OS Summary»

Раздел содержит информацию об ОС и ее настройках (Рисунок 32).



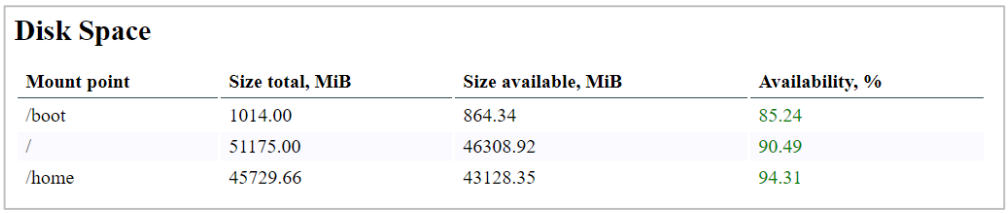
OS Summary

OS Distribution	Astra Linux 1.7 x86-64	
Kernel	5.4.0-54-generic (#astra31+ci49-Ubuntu SMP Mon Mar 15 18:57:33 MSK 2021)	
SELinux	disabled	
Service Manager	systemd	
Package Manager	apt	
Codepage	LANG	en_US.UTF-8
	LC_ADDRESS	
	LC_IDENTIFICATION	
	LC_MEASUREMENT	
	LC_MONETARY	
	LC_NAME	
	LC_NUMERIC	
	LC_PAPER	
	LC_TELEPHONE	
	LC_TIME	

Рисунок 32

3.11.2.2. Раздел «Disk Space»

Раздел содержит информацию о полном и доступном объеме дискового пространства для ППО (Рисунок 33).



Disk Space

Mount point	Size total, MiB	Size available, MiB	Availability, %
/boot	1014.00	864.34	85.24
/	51175.00	46308.92	90.49
/home	45729.66	43128.35	94.31

Рисунок 33

Описание назначения столбцов таблицы, а также информация о возможных значениях приведены в таблице (Таблица 21).

Таблица 21

Название столбца	Описание	Возможные значения (примеры значений)
Mount points	Каталог, к которому монтируется файловое хранилище (точка монтирования)	Путь к каталогу, например: `/home`
Size total, MiB	Размер файлового хранилища, примонтированного к заданному каталогу	Объем физической памяти в Мб, например: 45729,66
Size available, MiB	Объем свободного места в файловом хранилище, примонтированного к заданному каталогу	Объем физической памяти в Мб, например: 43128,35
Availability, %	Объем свободного места в файловом хранилище в процентном соотношении (к полному объему)	От 0 до 100, например: 94.31

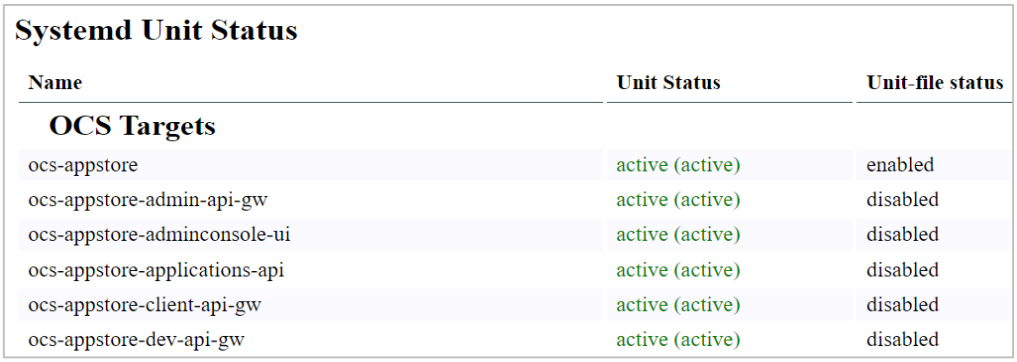
ПРИМЕЧАНИЕ. В случае если объем свободного места менее 15%, поле закрашивается цветом.

3.11.2.3. Раздел «Systemd Unit Status»

Радел содержит общую информацию о статусе сервисов ППО, компонентов среды функционирования ППО и инфраструктурных компонентов ППО и состоит из следующих подразделов:

3.11.2.3.1 OCS Targets

Подраздел содержит информацию о статусе конфигураций групп сервисов ППО (Рисунок 34).



Systemd Unit Status

Name	Unit Status	Unit-file status
OCS Targets		
ocs-appstore	active (active)	enabled
ocs-appstore-admin-api-gw	active (active)	disabled
ocs-appstore-adminconsole-ui	active (active)	disabled
ocs-appstore-applications-api	active (active)	disabled
ocs-appstore-client-api-gw	active (active)	disabled
ocs-appstore-dev-api-gw	active (active)	disabled

Рисунок 34

Описание назначения столбцов таблицы, а также информация о возможных значениях приведены в таблице (Таблица 22).

Таблица 22

Название столбца	Описание	Возможные значения (примеры значений)
Name	Имя конфигурации группы сервисов	Возможные значения определяются перечнем конфигураций групп сервисов ППО
Unit Status	Информация о статусе группы сервисов	`active` – группа сервисов запущена и выполняется; `activating` – группа сервисов запускается; `deactivating` – группа сервисов выключается; `inactive` – группа сервисов выключена; `failed` – при запуске группы сервисов произошла ошибка; `missed` – компонент отсутствует

Название столбца	Описание	Возможные значения (примеры значений)
Unit-file status	Информация о присутствии конфигурационного файла запуска группы сервисов в автозапуске	`enabled` – присутствует в автозапуске; `disabled` – отсутствует в автозапуске

3.11.2.3.2 OCS Services

Подраздел содержит информацию о статусе сервисов ППО (Рисунок 35).

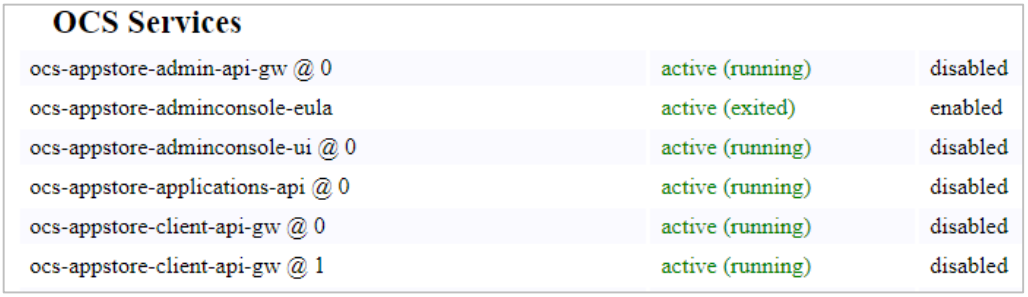


Рисунок 35

Описание назначения столбцов таблицы, а также информация о возможных значениях приведены в таблице (Таблица 23).

Таблица 23

Название столбца	Описание	Возможные значения (примеры значений)
Name	Название сервиса ППО	Возможные значения определяются перечнем сервисов ППО и имеют следующий формат `<имя группы сервисов>@<номер экземпляра сервиса в группе>.service`, например: `ocs-appstore-admin-api-gw@0.service`.
Unit Status	Информация о статусе сервиса	`active` – сервис запущен и выполняется; `activating` – сервис запускается; `deactivating` – сервис выключается; `inactive` – сервис выключен; `failed` – при запуске сервиса произошла ошибка
Unit-file status	Информация о присутствии конфигурационного файла запуска сервиса в автозапуске	`enabled` – присутствует в автозапуске; `disabled` – отсутствует в автозапуске

3.11.2.3.3 Mandatory services

Подраздел содержит информацию о статусе сервисов компонентов среды функционирования ППО и инфраструктурных компонентов ППО (Рисунок 36).

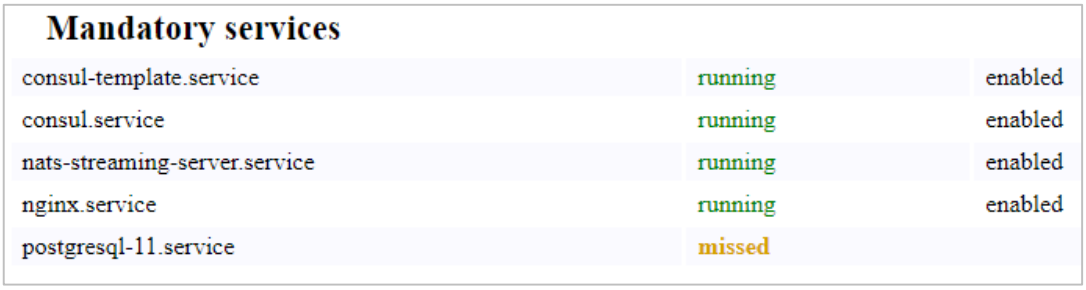


Рисунок 36

Описание назначения столбцов таблицы, а также информация о возможных значениях приведены в таблице (Таблица 24).

Таблица 24

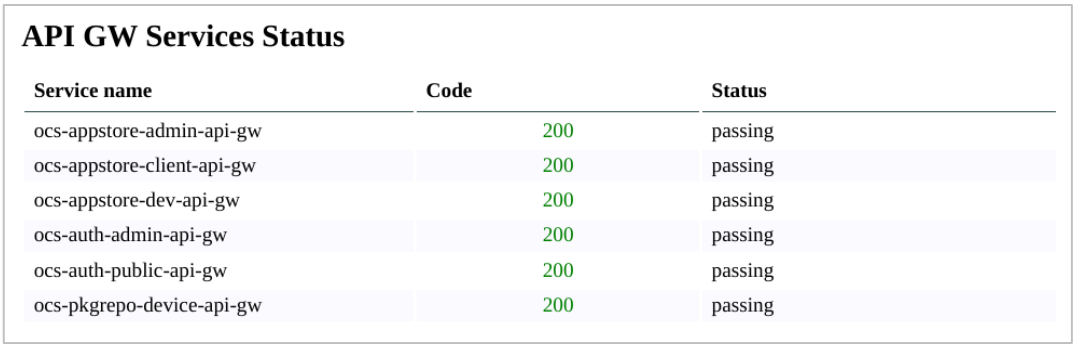
Название столбца	Описание	Возможные значения (примеры значений)

Name	Название сервиса компонента среды функционирования ППО или инфраструктурного компонента ППО	Возможные значения имеют следующий формат `<имя сервиса>.service` и определяются Разработчиком. Перечень возможных значений:
		`consul-template.service`
		`consul.service`
		`redpanda.service`
		`nginx.service`
		`postgresql-15.service`
		`postgresql.service`

Unit Status	Информация о статусе сервиса	`active` – сервис запущен и выполняется; `activating` – сервис запускается; `deactivating` – сервис выключается; `inactive` – сервис выключен; `failed` – при запуске сервиса произошла ошибка; `enabled` – сервис присутствует в автозапуске; `disabled` – сервис отсутствует в автозапуске; `missed` – компонент отсутствует
Unit-file status	Информация о присутствии конфигурационного файла запуска сервиса в автозапуске	`enabled` – присутствует в автозапуске; `disabled` – отсутствует в автозапуске

3.11.2.4. Раздел «API GW Service Status»

Раздел содержит информацию о статусе регистрации сервисов в системе обнаружения сервисов (Consul). На рисунке (Рисунок 37) приведен пример статуса регистрации сервисов в системе обнаружения сервисов.



API GW Services Status

Service name	Code	Status
ocs-appstore-admin-api-gw	200	passing
ocs-appstore-client-api-gw	200	passing
ocs-appstore-dev-api-gw	200	passing
ocs-auth-admin-api-gw	200	passing
ocs-auth-public-api-gw	200	passing
ocs-pkgrepo-device-api-gw	200	passing

Рисунок 37

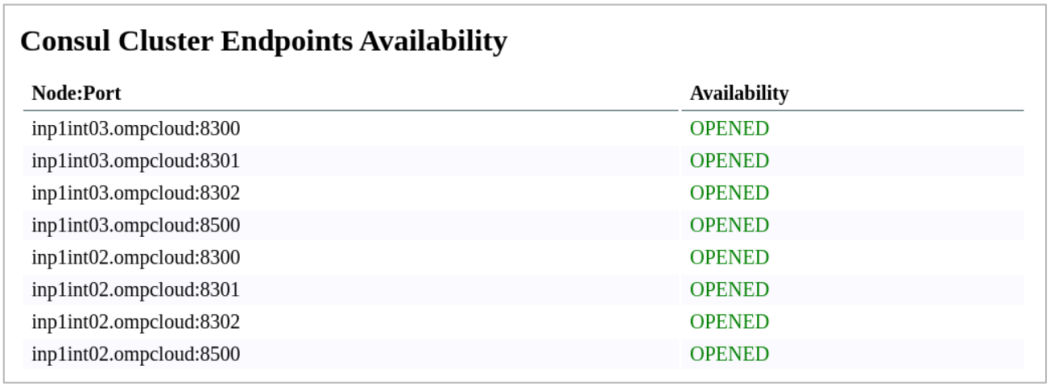
Описание назначения столбцов таблицы, а также информация о возможных значениях приведены в таблице (Таблица 25).

Таблица 25

Название столбца	Описание	Возможные значения (примеры значений)
Service name	Название сервиса ППО	Возможные значения определяются перечнем сервисов ППО
Code	Код http-ответа	Возможные значения определяются протоколом HTTP
Status	Информация о статусе регистрации сервиса в системе обнаружения сервисов (Consul)	Возможные значения определяются Consul. Статус «passing» означает, что проверка пройдена успешно

3.11.2.5. Раздел «Consul Cluster Endpoints Availability»

Раздел содержит информацию о проверке доступности интерфейсных функций системы обнаружения сервисов (Consul). На рисунке (Рисунок 38) приведен пример отображения информации о доступности интерфейсных функций системы обнаружения сервисов.



Consul Cluster Endpoints Availability

Node:Port	Availability
inp1int03.ompcloud:8300	OPENED
inp1int03.ompcloud:8301	OPENED
inp1int03.ompcloud:8302	OPENED
inp1int03.ompcloud:8500	OPENED
inp1int02.ompcloud:8300	OPENED
inp1int02.ompcloud:8301	OPENED
inp1int02.ompcloud:8302	OPENED
inp1int02.ompcloud:8500	OPENED

Рисунок 38

Перечень интерфейсных функций Consul приведен в документации на Consul (https://www.consul.io/docs/install/ports). Информация о доступности интерфейсных функций Consul предоставляется только в случае кластерной (многонодовой) конфигурации.

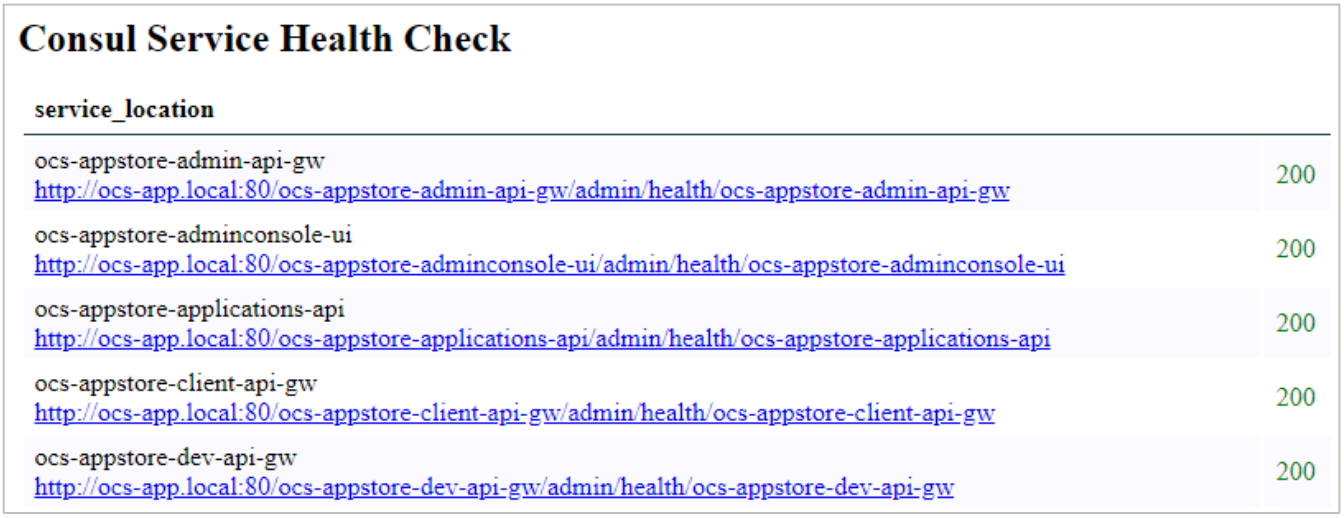
Описание назначения столбцов таблицы, а также информация о возможных значениях приведены в таблице (Таблица 26).

Таблица 26

Название столбца	Описание	Возможные значения (примеры значений)
Node:Port	Адрес функции	Адрес функции представлен в следующем формате: `<имя хоста>:<порт>`. Проверка выполняется только для функций, доступных на следующих портах: 8300, 8301, 8302, 8500. Например: `acenter.example:8300`
Availability	Статус доступности функции	В случае доступности функции принимает значение «OPENED». В ином случае выводится код ошибки и сообщение, определяемое Consul

3.11.2.6. Раздел «Consul Service Health Check»

Раздел содержит информацию о статусе регистрации сервисов ППО в системе обнаружения сервисов Consul (Рисунок 39).



Consul Service Health Check

service_location	
ocs-appstore-admin-api-gw http://ocs-app.local:80/ocs-appstore-admin-api-gw/admin/health/ocs-appstore-admin-api-gw	200
ocs-appstore-adminconsole-ui http://ocs-app.local:80/ocs-appstore-adminconsole-ui/admin/health/ocs-appstore-adminconsole-ui	200
ocs-appstore-applications-api http://ocs-app.local:80/ocs-appstore-applications-api/admin/health/ocs-appstore-applications-api	200
ocs-appstore-client-api-gw http://ocs-app.local:80/ocs-appstore-client-api-gw/admin/health/ocs-appstore-client-api-gw	200
ocs-appstore-dev-api-gw http://ocs-app.local:80/ocs-appstore-dev-api-gw/admin/health/ocs-appstore-dev-api-gw	200

Рисунок 39

Описание назначения столбцов таблицы, а также информация о возможных значениях приведены в таблице (Таблица 27).

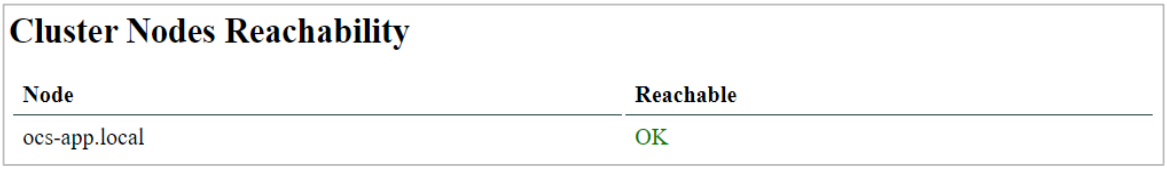
Таблица 27

Название столбца	Описание	Возможные значения (примеры значений)
Первый столбец	Название сервиса ППО и URL-адрес функции (endpoint) сервиса «healthcheck»	Возможные значения определяются перечнем сервисов ППО
Второй столбец	Код http-ответа	Возможные значения определяются протоколом HTTP

Перечисленные заголовки `"service_location"`, `"expose_location"`, `"service_vhost"`, `"expose_port"`, `"static"` – режимы работы `consul-template`.

3.11.2.7. Раздел «Cluster Nodes Reachability»

Раздел содержит информацию о результатах проверки доступности серверов (нод) кластера (Рисунок 40).



Cluster Nodes Reachability

Node	Reachable
ocs-app.local	OK

Рисунок 40

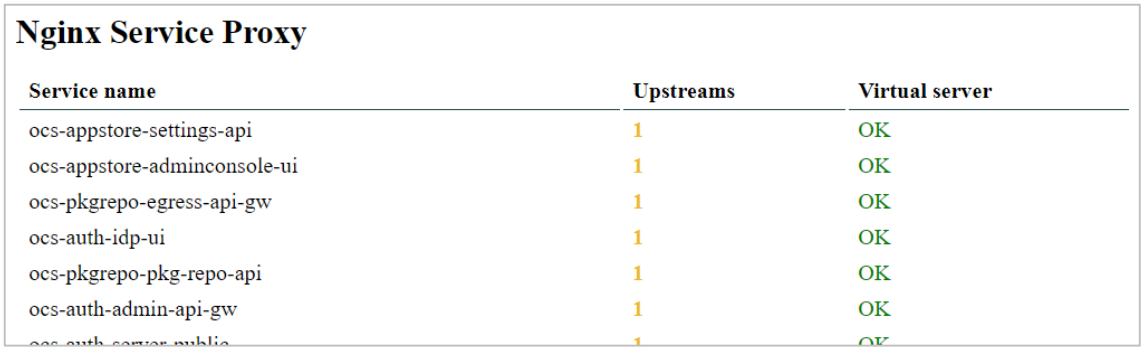
Описание назначения столбцов таблицы, а также информация о возможных значениях приведены в таблице (Таблица 28).

Таблица 28

Название столбца	Описание	Возможные значения (примеры значений)
Node	Адрес сервера (хоста)	Определяется доменными именами хостов
Reachable	Информация о доступности сервера	Может принимать значения: «OK» (в случае доступности) или содержать сообщение об ошибке, которое вернет утилита `ping`

3.11.2.8. Раздел «Nginx Service Proxy»

Раздел содержит информацию о проверке конфигурации балансировщика сервисов Nginx Web Server для каждого сервиса ППО (Рисунок 41).



Nginx Service Proxy

Service name	Upstreams	Virtual server
ocs-appstore-settings-api	1	OK
ocs-appstore-adminconsole-ui	1	OK
ocs-pkgrepo-egress-api-gw	1	OK
ocs-auth-idp-ui	1	OK
ocs-pkgrepo-pkg-repo-api	1	OK
ocs-auth-admin-api-gw	1	OK
ocs-auth-server-public	1	OK

Рисунок 41

Описание назначения столбцов таблицы, а также информация о возможных значениях приведены в таблице (Таблица 29).

Таблица 29

Название столбца	Описание	Возможные значения (примеры значений)
Service name	Название сервиса ППО	Возможные значения определяются перечнем сервисов ППО
Upstreams	Количество экземпляров сервиса, заданных в конфигурационном файле Nginx	Целочисленные значения от 1 до n

Название столбца	Описание	Возможные значения (примеры значений)
Virtual server	Информация о наличии секции «server» для указанного сервиса ППО в конфигурационном файле Nginx. В данной секции заданы настройки «виртуального» сервиса ППО, который осуществляет перенаправление (проксирование) http-запросов на «реальные» экземпляры сервиса	«ОК» – секция `server` присутствует; «No server block found!» – секция отсутствует

3.11.2.9. Раздел «Filestorage Configuration»

Раздел содержит информацию о конфигурации файловых хранилищ ПМ и ПООС (Рисунок 42).



Рисунок 42

Настройка «Filestorage location» содержит путь к каталогу и его статус.

В настройке «Configuration file» указан конфигурационный файл, в котором задан путь к файловому хранилищу.

3.12. Самостоятельная установка и настройка СУБД Postgres Pro и СУБД PostgreSQL 14/15/16/17

3.12.1. Установить на серверы БД необходимые пакеты согласно п. 3.10.8.

3.12.2. Установить и инициализировать СУБД.

При инициализации СУБД необходимо установить следующие значения параметров:

```
LC_COLLATE 'en_US.UTF-8'
LC_CTYPE 'en_US.UTF-8'
ENCODING UTF8
```

Установка и инициализация СУБД осуществляется в соответствии с ЭД на СУБД.

ПРИМЕЧАНИЕ. В рамках установки Postgres Pro обязательно должны быть установлены следующие пакеты:

- `postgrespro-<std|ent>-<версия>`;
- `postgrespro-<std|ent>-<версия>-client`;
- `postgrespro-<std|ent>-<версия>-contrib`;
- `postgrespro-<std|ent>-<версия>-libs`;
- `postgrespro-<std|ent>-<версия>-server`.

3.12.3. Создать суперпользователя с помощью скрипта `samples/sql/create_superuser.sql`, выполнив команду:

```
psql -U <пользователь, от имени которого выполняется команда> -h
<адрес хоста СУБД> -f create_superuser.sql -v login='<имя
суперпользователя>' -v pass='<пароль суперпользователя>' -v
expr='<срок действия учетной записи суперпользователя>'
```

Например:

```
psql -U postgres -h 192.168.137.15 -f create_superuser.sql -v
login='ompdbuser' -v pass='Admin123!' -v expr='10 years'
```

3.12.4. Назначить пароль для пользователя `postgres` с помощью следующих команд:

```
psql -U postgres
ALTER USER postgres with PASSWORD 'пароль';
exit
```

3.12.5. В конфигурационных файлах СУБД `pg_hba.conf` и `postgresql.conf` задать следующие параметры:

- тип соединения, диапазон IP-адресов клиентов БД;
- имя БД, имя пользователя;
- способ аутентификации клиентов;
- пароль пользователя СУБД в параметре `pg_superuser_password`.

3.12.6. Установить расширения `pg_partman` и `pg_cron` с помощью команд:

– ОС Альт Сервер 10 и СУБД PostgreSQL 14:

```
sudo rpm -ivh postgresql14-pg_cron-1.6.2-1.alt.p10.x86_64.rpm
tar -xf pg_partman--5.1.0.tar.gz -C /usr/share/pgsql/extension
```

– ОС Альт Сервер 10 и СУБД PostgreSQL 15:

```
sudo rpm -ivh postgresql15-pg_cron-1.6.2-1.alt.p10.x86_64.rpm
tar -xf pg_partman--5.1.0.tar.gz -C /usr/share/pgsql/extension
```

– ОС Альт Сервер 10 и СУБД PostgreSQL 16:

```
sudo rpm -ivh postgresql16-pg_cron-1.6.2-1.alt.p10.x86_64.rpm
tar -xf pg_partman--5.1.0.tar.gz -C /usr/share/pgsql/extension
```

– ОС Альт Сервер 10 и СУБД PostgreSQL 17:

```
sudo rpm -ivh postgresql17-pg_cron-1.6.7-alt1.x86_64.rpm
tar -xf pg_partman--5.1.0.tar.gz -C /usr/share/pgsql/extension
```

– ОС Альт 8 СП релиз 10 и СУБД PostgreSQL 15:

```
sudo rpm -ivh postgresql15-pg_cron-1.6.2-1.alt.p10.x86_64.rpm
tar -xf pg_partman--5.1.0.tar.gz -C /usr/share/pgsql/extension
```

– ОС Альт 8 СП релиз 10 и СУБД PostgreSQL 16:

```
sudo rpm -ivh postgresql16-pg_cron-1.6.2-1.alt.p10.x86_64.rpm
tar -xf pg_partman--5.1.0.tar.gz -C /usr/share/pgsql/extension
```

– ОС Альт 8 СП релиз 10 и СУБД Postgres Pro 16:

```
sudo rpm -ivh pg_cron_16pro-std-cert-1.6.2-alt10.x86_64.rpm
tar -xf pg_partman--5.1.0.tar.gz -C /opt/pgpro/std-16/share/extension
```

– ОС Альт 8 СП релиз 10 и СУБД Postgres Pro 17:

```
sudo rpm -ivh pg_cron_17pro-std-cert-1.6.3-alt10.x86_64.rpm
tar -xf pg_partman--5.1.0.tar.gz -C /opt/pgpro/std-16/share/extension
```

– ОС Альт 8 СП релиз 10 и СУБД Postgres Pro 15 Cert:

```
sudo rpm -ivh pg_cron_15pro-std-cert-1.6.2-alt10.x86_64.rpm
tar -xf pg_partman--5.1.0.tar.gz -C /opt/pgpro/std-15/share/extension
```

– ОС РЕД ОС 7.3 и СУБД PostgreSQL 14:

```
sudo rpm -ivh pg_cron_14-1.6.2-1.redos7.x86_64.rpm
tar -xf pg_partman--5.1.0.tar.gz -C /usr/pgsql-14/share/extension
```

– ОС РЕД ОС 7.3 и СУБД PostgreSQL 15:

```
sudo rpm -ivh pg_cron_15-1.6.2-1.redos7.x86_64.rpm
tar -xf pg_partman--5.1.0.tar.gz -C /usr/pgsql-15/share/extension
```

– ОС РЕД ОС 7.3 и СУБД PostgreSQL 16:

```
sudo rpm -ivh pg_cron_16-1.6.2-1.redos7.x86_64.rpm
tar -xf pg_partman--5.1.0.tar.gz -C /usr/pgsql-16/share/extension
```

– ОС РЕД ОС 7.3 и СУБД PostgreSQL 17:

```
sudo rpm -ivh pg_cron_17-1.6.4-1.redos7.x86_64.rpm
tar -xf pg_partman--5.1.0.tar.gz -C /usr/pgsql-16/share/extension
```

– ОС РЕД ОС 7.3 и СУБД Postgres Pro 15:

```
sudo rpm -ivh pg_cron_15pro-std-1.6.2-1.redos7.x86_64.rpm
tar -xf pg_partman--5.1.0.tar.gz -C /opt/pgpro/std-15/share/extension
```

– ОС РЕД ОС 7.3 и СУБД Postgres Pro 16:

```
sudo rpm -ivh pg_cron_16pro-std-1.6.2-1.redos7.x86_64.rpm
tar -xf pg_partman--5.1.0.tar.gz -C /opt/pgpro/std-16/share/extension
```

– ОС РЕД ОС 7.3 и СУБД Postgres Pro 17:

```
sudo rpm -ivh pg_cron_17pro-std-1.6.3-1.redos7.x86_64.rpm
tar -xf pg_partman--5.1.0.tar.gz -C /opt/pgpro/std-16/share/extension
```

– ОС РЕД ОС 7.3 и СУБД Postgres Pro 14 Cert:

```
sudo rpm -ivh pg_cron_14pro-std-cert-1.6.2-1.redos7.x86_64.rpm
tar -xf pg_partman--5.1.0.tar.gz -C /opt/pgpro/std-14/share/extension
```

– ОС РЕД ОС 7.3 и СУБД Postgres Pro 15 Cert:

```
sudo rpm -ivh pg_cron_15pro-std-1.6.2-1.redos7.x86_64.rpm
tar -xf pg_partman--5.1.0.tar.gz -C /opt/pgpro/std-15/share/extension
```

– ОС РЕД ОС 8 и СУБД PostgreSQL 16:

```
sudo rpm -ivh pg_cron_16-1.6.2-1.red80.x86_64.rpm
tar -xf pg_partman--5.1.0.tar.gz -C /usr/pgsql-16/share/extension
```

– ОС РЕД ОС 8 и СУБД PostgreSQL 17:

```
sudo rpm -ivh pg_cron_17-1.6.3-1.red80.x86_64.rpm
tar -xf pg_partman--5.1.0.tar.gz -C /usr/pgsql-16/share/extension
```

– ОС РЕД ОС 8 и СУБД Postgres Pro 16:

```
sudo rpm -ivh pg_cron_16pro-std-1.6.2-1.redos7.x86_64.rpm
tar -xf pg_partman--5.1.0.tar.gz -C /opt/pgpro/std-16/share/extension
```

– ОС РЕД ОС 8 и СУБД Postgres Pro 17:

```
sudo rpm -ivh pg_cron_17pro-std-1.6.3-1.redos7.x86_64.rpm
tar -xf pg_partman--5.1.0.tar.gz -C /opt/pgpro/std-16/share/extension
```

– ОС Astra Linux SE 1.7 и СУБД PostgreSQL 14:

```
sudo rpm -ivh pg-cron_1.6.2_14_smolensk.amd64.deb
tar -xf pg_partman--5.1.0.tar.gz -C /usr/pgsql-14/share/extension
```

– ОС Astra Linux SE 1.7 и СУБД PostgreSQL 15:

```
sudo rpm -ivh pg-cron_1.6.2_15_smolensk.amd64.deb
tar -xf pg_partman--5.1.0.tar.gz -C /usr/pgsql-15/share/extension
```

– ОС Astra Linux SE 1.7 и СУБД PostgreSQL 16:

```
sudo dpkg -i pg-cron_1.6.2_16_smolensk.amd64.deb
tar -xf pg_partman--5.1.0.tar.gz -C /usr/pgsql-16/share/extension
```

– ОС Astra Linux SE 1.7 и СУБД PostgreSQL 17:

```
sudo dpkg -i pg-cron_1.6.3_17_smolensk.amd64.deb
tar -xf pg_partman--5.1.0.tar.gz -C /usr/pgsql-16/share/extension
```

– ОС Astra Linux SE 1.7 и СУБД Postgres Pro 14 Cert:

```
sudo dpkg -i pg-cron_1.6.2_pro-std-cert-14_smolensk.amd64.deb
tar -xf pg_partman--5.1.0.tar.gz -C /opt/pgpro/std-14/share/extension
```

– ОС Astra Linux SE 1.7 и СУБД Postgres Pro 15 Cert:

```
sudo dpkg -i pg-cron_1.6.2_15_smolensk-pro-std-cert.amd64.deb
tar -xf pg_partman--5.1.0.tar.gz -C /opt/pgpro/std-15/share/extension
```

– ОС Astra Linux SE 1.7 и СУБД Postgres Pro 16 Cert:

```
sudo dpkg -i pg-cron_1.6.2_16_smolensk-pro-std-cert.amd64.deb
tar -xf pg_partman--5.1.0.tar.gz -C /opt/pgpro/std-15/share/extension
```

– ОС Astra Linux SE 1.8 и СУБД PostgreSQL 16:

```
sudo rpm -ivh pg-cron_1.6.2_16_smolensk.amd64.deb
tar -xf pg_partman--5.1.0.tar.gz -C /usr/pgsql-16/share/extension
```

– ОС Astra Linux SE 1.8 и СУБД PostgreSQL 17:

```
sudo rpm -ivh pg-cron_1.6.3_17_smolensk.amd64.deb
tar -xf pg_partman--5.1.0.tar.gz -C /usr/pgsql-16/share/extension
```

– ОС Astra Linux SE 1.8 и СУБД Postgres Pro 16 Cert:

```
sudo rpm -ivh pg-cron_1.6.2_16_smolensk-pro-std-cert.amd64.deb
tar -xf pg_partman--5.1.0.tar.gz -C /opt/pgpro/std-16/share/extension
```

– ОС Astra Linux SE 1.8 и СУБД Postgres Pro 17 Cert:

```
sudo rpm -ivh pg-cron_1.6.3_17_smolensk-pro-std-cert.amd64.deb
tar -xf pg_partman--5.1.0.tar.gz -C /opt/pgpro/std-16/share/extension
```

– ОС Ubuntu версии 22.04 и СУБД PostgreSQL 14:

```
sudo rpm -ivh postgresql-14-cron-1.6.2-1.jammy.amd64.deb
tar -xf pg_partman--5.1.0.tar.gz -C /usr/share/postgresql/14/extension
```

– ОС Ubuntu версии 22.04 и СУБД PostgreSQL 15:

```
sudo rpm -ivh postgresql-15-cron-1.6.2-1.jammy.amd64.deb
tar -xf pg_partman--5.1.0.tar.gz -C /usr/share/postgresql/15/extension
```

– ОС Ubuntu версии 22.04 и СУБД PostgreSQL 16:

```
sudo rpm -ivh postgresql-16-cron-1.6.2-1.jammy.amd64.deb
tar -xf pg_partman--5.1.0.tar.gz -C /usr/share/postgresql/16/extension
```

– ОС Ubuntu версии 22.04 и СУБД PostgreSQL 17:

```
sudo rpm -ivh postgresql-17-cron-1.6.3-1.jammy.amd64.deb
tar -xf pg_partman--5.1.0.tar.gz -C /usr/share/postgresql/16/extension
```

– ОС Ubuntu версии 22.04 и СУБД Postgres Pro 16:

```
sudo rpm -ivh postgresql-16pro-cron-1.6.2-1.jammy.amd64.deb
tar -xf pg_partman--5.1.0.tar.gz -C /opt/pgpro/std-16/share/extension
```

– ОС Ubuntu версии 22.04 и СУБД Postgres Pro 17:

```
sudo rpm -ivh postgresql-17pro-cron-1.6.3-1.jammy.amd64.deb
tar -xf pg_partman--5.1.0.tar.gz -C /opt/pgpro/std-16/share/extension
```

– ОС Ubuntu версии 24.04 и СУБД PostgreSQL 16:

```
sudo rpm -ivh postgresql-16-cron_1.6.5-1.pgdg24.04+1_amd64.deb
tar -xf pg_partman--5.1.0.tar.gz -C /usr/share/postgresql/16/extension
```

– ОС Ubuntu версии 24.04 и СУБД PostgreSQL 17:

```
sudo rpm -ivh postgresql-17-cron-1.6.3-1.noble.amd64.deb
tar -xf pg_partman--5.1.0.tar.gz -C /usr/share/postgresql/16/extension
```

– ОС Debian версии 12 и СУБД PostgreSQL 14:

```
sudo rpm -ivh postgresql-14-cron-1.6.2-1.bookworm.amd64.deb
tar -xf pg_partman--5.1.0.tar.gz -C /usr/share/postgresql/14/extension
```

– ОС Debian версии 12 и СУБД PostgreSQL 15:

```
sudo rpm -ivh postgresql-15-cron-1.6.2-1.bookworm.amd64.deb
tar -xf pg_partman--5.1.0.tar.gz -C /usr/share/postgresql/15/extension
```

– ОС Debian версии 12 и СУБД PostgreSQL 16:

```
sudo rpm -ivh postgresql-16-cron-1.6.2-1.bookworm.amd64.deb
tar -xf pg_partman--5.1.0.tar.gz -C /usr/share/postgresql/16/extension
```

– ОС Debian версии 12 и СУБД PostgreSQL 17:

```
sudo rpm -ivh postgresql-17-cron-1.6.3-1.bookworm.amd64.deb
tar -xf pg_partman--5.1.0.tar.gz -C /usr/share/postgresql/16/extension
```

– ОС Debian версии 12 и СУБД Postgres Pro 16:

```
sudo rpm -ivh postgresql-16pro-cron-1.6.2-1.bookworm.amd64.deb
tar -xf pg_partman--5.1.0.tar.gz -C /opt/pgpro/std-16/share/extension
```

– ОС Debian версии 12 и СУБД Postgres Pro 17:

```
sudo rpm -ivh postgresql-17pro-cron-1.6.3-1.bookworm.amd64.deb
tar -xf pg_partman--5.1.0.tar.gz -C /opt/pgpro/std-16/share/extension
```

– Platform V SberLinux OS Server 9.2 и СУБД Platform V Pangolin DB 6:

```
tar -xf pg_partman--5.2.4.tar.gz -C /usr/pangolin/share/extension
```

Указанные RPM-пакеты и архивы находятся на DVD с загрузочными модулями ППО в архиве `/server/AC_<версия ППО>_install-infra.tar.gz/install-infra/binary/postgresql/`.

3.12.7. Перезапустить сервис СУБД в соответствии с документацией на СУБД.

4. УПРАВЛЕНИЕ КОМПОНЕНТАМИ СРЕДЫ ФУНКЦИОНИРОВАНИЯ ППО, ИНФРАСТРУКТУРНЫМИ КОМПОНЕНТАМИ ППО, СЕРВИСАМИ, НАСТРОЙКАМИ СЕРВИСОВ И ПОДСИСТЕМ

4.1. Управление компонентами среды функционирования ППО и инфраструктурными компонентами ППО

Управление компонентами среды фунционирования ППО и инфраструктурными компонентами ППО заключается в их установке, обновлении и удалении и осуществляется с помощью скрипта `deploy-infra.sh` из каталога `install-<версия ППО>`, созданного на этапе развертывания управляющей ЭВМ (подраздел 3.3).

Формат команды управления сервисами имеет следующий вид:

```
ANSIBLE_USER="<имя пользователя>" ./deploy-infra.sh <параметры>
```

Описание параметров команды управления:

1) `<имя пользователя>`

В параметре указывается имя привилегированного sudo-пользователя, под которым настроен SSH доступ к серверам приложений, серверам БД и контент-серверам;

2) `-A, --action`

Данный параметр задает действие, которое необходимо выполнить. Перечень допустимых действий и соответствующие им значения параметра приведены в таблице (Таблица 30).

Таблица 30

Значение параметра «--action»	Действие
`deploy`	Установка
`start`	Запуск
`stop`	Остановка
`flush_all`	Удаление

По умолчанию параметр (если не задано иное значение) имеет значение deploy;

3) `-c, --components`

Данный параметр задает компонент среды функционирования ППО или инфраструктурный компонент ППО, для которого будет выполнена команда управления и может принимать следующие значения:

- `dnsmasq;`
- `nginx;`
- `consul;`
- `consul-content;`
- `consul-template;`
- `redpanda;`
- `valkey;`
- `ocs-user;`
- `db;`
- `syncthing.`

В данном параметре может задаваться список подсистем, например:

```
--components dnsmasq, nginx
```

По умолчанию (если параметр на задан) команда управления будет применена ко всем компонентам;

4) `-d, --database`

Перечень допустимых значений параметра приведен в таблице (см. Таблица 14).

По умолчанию (если параметр не задан) будет использоваться значение, заданное в параметре `pg_version` конфигурационного файла `config/vars/_vars.yml`.

При отсутствии СУБД в перечне компонентов (параметр `--components`) значение данного параметра будет игнорироваться;

5) `--skip-database`

При наличии данного параметра СУБД не устанавливается;

6) `-l, --limit`

Данный параметр задает перечень хостов, для которых будет выполнена команда управления, например:

```
--limit example01.omp,example02.omp
```

По умолчанию (если параметр на задан) команда управления будет применена ко всем хостам согласно инвентарному файлу `inventories/hosts.yml`;

7) `-x, --extra-vars`

В данном параметре передаются внешние переменные для скриптов развертывания. В ППО используются следующие внешние переменные:

– `pg_uninstall_delete_data=true` – служит для удаления данных при удалении СУБД PostgreSQL;

8) `--force-infra-install`

Флаг служит для управления принудительной повторной установкой компонентов среды функционирования и инфраструктурных компонентов ППО, в случаях, когда версия компонентов не изменилась и может принимать следующие значения:

– `false` – повторная установка компонентов той же версии выполнена не будет;

– `true` – будет выполнена повторная установка компонентов независимо от того изменилась версия или нет.

По умолчанию (если флаг не задан) флаг имеет значение `true`;

9) `-e, --env`

Данный параметр задает окружение, для которого необходимо выполнить настройку и установку компонентов. Более подробная информация по настройке ППО для установки его на различные окружения приведена в п. 3.10.16;

10) `--help`

Вывод справочной информации.

Примеры команд управления:

1) Установка или обновление всех компонентов:

```
ANSIBLE_USER="omp" ./deploy-infra.sh --database 12
```

2) Установка или обновление Nginx на хосте `ocs-app.local`:

```
ANSIBLE_USER="omp" ./deploy-infra.sh --components nginx --limit ocs-app.local
```

3) Удаление Nginx:

```
ANSIBLE_USER="omp" ./deploy-infra.sh --components nginx --action flush_all
```

4) Удаление СУБД PostgreSQL (с удалением данных):

```
ANSIBLE_USER="<имя пользователя>" ./deploy-infra.sh  --components db --action flush_all --extra-vars "pg_uninstall_delete_data=true"
```

5) Получение справочной информации:

```
./deploy-infra.sh --help
```

4.2. Управление сервисами ППО

Управление сервисами ППО заключается в их установке, запуске, остановке, перезапуске, изменении настроек и осуществляется с помощью скрипта `deploy-ac.sh` из каталога `install-<версия ППО>`, созданного на этапе развертывания управляющей ЭВМ (подраздел 3.3).

Формат команды управления сервисами имеет следующий вид:

```
ANSIBLE_USER="<имя пользователя>" ./deploy-ac.sh <параметры>
```

Описание параметров команды управления:

1) `<имя пользователя>`

В параметре указывается имя привилегированного sudo-пользователя, под которым настроен SSH доступ к серверам приложений, серверам БД и контент-серверам;

2) `-s, --subsystems`

Данный параметр задает подсистему, для которой будет выполнена команда управления, и может принимать следующие значения:

- `auth` (для ПБ);
- `appstore` (для ПМ);
- `emm` (для ПУ);
- `mt` (для ПУТ);
- `push` (для ПСУ);
- `cdn` (для CDN);
- `pkgrepo` (для ПООС).

В данном параметре может задаваться список подсистем, например:

```
--subsystems auth,appstore,pkgrepo,emm,mt,cdn,push
```

По умолчанию параметр (если иное значение не задано) имеет значение:

```
--subsystems auth,appstore,pkgrepo,emm,mt,push
```

3) `-a, --apps`

Данный параметр задает перечень сервисов, для которых будет выполнена команда управления. Например:

```
--apps ocs-auth-adminconsole-ui,ocs-appstore-adminconsole-ui
```

Если необходимо выполнить команду сразу для всех сервисов подсистемы, потребуется перечислить через запятую все сервисы подсистемы либо задать значение параметра:

```
--apps all
```

По умолчанию параметр (если иное значение не задано) имеет значение `all`.

В случае если заданные в параметре `--apps` сервисы не соответствуют заданным в параметре `--subsystems` подсистемам, управляющая команда к таким сервисам применена не будет. При этом управление шлюзами доступа (сервисами шлюзов доступа) осуществляется в рамках той подсистемы, для которой они предназначены. Состав подсистем приведен в таблице (Таблица 31).

Таблица 31

Значение параметра «--subsystems»	Сервисы (значение параметра «--apps»)
ПБ	
auth	ocs-auth-admin-api-gw
	ocs-auth-public-api-gw
	ocs-auth-admin-cross-tenant-api-gw
	ocs-auth-server-public-proxy
	ocs-auth-idp-api
	ocs-auth-accounts-devices-api
	ocs-auth-accounts-users-api
	ocs-auth-server-admin
	ocs-auth-server-public
	ocs-auth-audit-api
	ocs-auth-subsystems-api
	ocs-auth-config-api
	ocs-auth-adminconsole-ui
	ocs-auth-idp-ui
ПМ	
appstore	ocs-appstore-applications-api
	ocs-appstore-settings-api
	ocs-appstore-adminconsole-ui
	ocs-appstore-devconsole-ui
	ocs-appstore-admin-api-gw
	ocs-appstore-client-api-gw
	ocs-appstore-dev-api-gw
	ocs-appstore-egress-api-gw
ПУ	
emm	ocs-emm-applications-api
	ocs-emm-dispatcher-api
	ocs-emm-devices-api
	ocs-emm-state-manager-api
	ocs-emm-enrollments-api
	ocs-emm-policies-api
	ocs-emm-users-api
	ocs-emm-journal-api
	ocs-emm-jobs-api
	ocs-emm-locations-api

Значение параметра «--subsystems»	Сервисы (значение параметра «--apps»)
	ocs-emm-admin-api-gw
	ocs-emm-device-api-gw
	ocs-emm-egress-api-gw
ПУТ	
mt	ocs-mt-tenants-api
	ocs-mt-organizations-api
	ocs-mt-admin-api-gw
	ocs-mt-egress-api-gw
ПСУ	
push	ocs-push-main-api
	ocs-push-transport
	ocs-push-admin-api-gw
	ocs-push-public-api-gw
	ocs-push-egress-api-gw
ПООС	
pkgrepo	ocs-pkgrepo-pkg-repo-api
	ocs-pkgrepo-device-api-gw
	ocs-pkgrepo-admin-api-gw
	ocs-pkgrepo-egress-api-gw
CDN	
cdn	ocs-cdn-admin-api-gw
	ocs-cdn-mobile-api-gw

4) `-A, --action`

Данный параметр задает действие, которое необходимо выполнить. Перечень допустимых действий и соответствующие им значения параметра приведены в таблице (Таблица 32).

Таблица 32

Значение параметра «--action»	Действие
deploy	Установка
start	Запуск
stop	Остановка
restart	Перезапуск
config	Изменение настроек (переустановка конфигурационного файла)
flush_all	Удаление

По умолчанию параметр (если не задано иное значение) имеет значение `deploy`.

ВНИМАНИЕ! Установка подсистем ППО должна осуществляться строго в следующей последовательности: ПБ, ПМ, ПООС, ПУ, ПУТ, CDN, ПСУ;

5) `-C, --clients`

Данный параметр задает OIDC клиентов, для которых будет выполнена команда управления.

Например:

```
--clients auth-admin-console, aps-admin-console
```

При необходимости выполнить команду сразу для всех OIDC клиентов потребуется перечислить через запятую все OIDC клиенты либо задать значение параметра:

```
--clients all
```

По умолчанию параметр (если не задано иное значение) имеет значение `all`;

6) `-d, --database`

Данный параметр задает СУБД, которая установлена на сервере БД. Перечень допустимых значений параметра приведен в таблице (см. Таблица 14).

Например:

```
--database 12
```

По умолчанию (если параметр не задан) будет использоваться значение, заданное в параметре `pg_version` конфигурационного файла `config/vars/_vars.yml`;

7) `-e, --env`

Данный параметр задает окружение, для которого необходимо выполнить настройку и установку ППО. Более подробная информация по настройке ППО для установки его на различные окружения приведена в п. 3.10.16;

8) `--help`

Вывод справочной информации.

Примеры команд управления:

1) Остановка всех сервисов ПМ:

```
ANSIBLE_USER="omp" ./deploy-ac.sh --action stop
```

2) Запуск сервисов `ocs-appstore-applications-api` и `ocs-appstore-adminconsole` ПМ:

```
ANSIBLE_USER="omp" ./deploy-ac.sh --subsystems appstore --apps ocs-appstore-applications-api,ocs-appstore-adminconsole --action start
```

3) Получение справочной информации:

```
./deploy-ac.sh --help
```

4.3. Управление настройками сервисов и подсистем ППО

Управление настройками сервисов и подсистем ППО может осуществляться 2 способами.

4.3.1. Способ 1 (рекомендуемый)

4.3.1.1. Задать требуемые значения параметров в конфигурационных файлах сценариев установки ППО и подсистем ППО.

4.3.1.2. Переустановить конфигурационные файлы с помощью команды:

```
ANSIBLE_USER="<имя пользователя>" ./deploy-ac.sh --action config
```

Подробное описание параметров запуска скрипта `deploy-ac.sh` приведено в подразделе 4.2.

4.3.2. Способ 2

4.3.2.1. Задать требуемые значения параметров в конфигурационных файлах сервисов и подсистем ППО. Описание параметров конфигурационных файлов сценариев установки подсистем ППО приведено в разделе 13.

4.3.2.2. Перезапустить требуемые сервисы с помощью команды:

```
ANSIBLE_USER="<имя пользователя>" ./deploy-ac.sh --subsystems <идентификатор подсистемы> --apps <перечень сервисов подсистемы> --action restart
```

Подробное описание параметров запуска скрипта `deploy-ac.sh` приведено в подразделе 4.2.

5. РЕЗЕРВНОЕ КОПИРОВАНИЕ

ВНИМАНИЕ! Приведенные ниже имена файлов и каталогов характерны для типового варианта установки ППО и среды функционирования ППО.

5.1. Резервное копирование после установки (обновления) ППО

После успешной установки (обновления) ППО необходимо создать резервную копию каталога `install-<версия ППО>/install-ac-mt/`.

5.2. Периодическое резервное копирование и резервное копирование перед установкой обновлений

Периодичность резервного копирования определяется регламентами эксплуатирующей организации.

Периодическое резервное копирование и резервное копирование перед установкой обновлений выполняется в приведенной ниже последовательности.

Подробная информация об особенностях резервного копирования ППО приведена в документе «Рекомендации по резервному копированию».

6. ОБНОВЛЕНИЕ ППО И ОС АВРОРА

6.1. Порядок обновления

В рамках поддержки жизненного цикла ППО предприятие-изготовитель вносит в него изменения, направленные на улучшение эксплуатационных характеристик и устранение недостатков.

Доведение информации о выпуске обновлений ППО до каждого потребителя ППО осуществляется посредством отправки сообщений на электронные адреса потребителей и/или посредством публикации на официальном веб-сайте предприятия-разработчика (https://www.omp.ru, https://auroraos.ru, https://cve.omp.ru).

Предусмотрены следующие способы предоставления обновлений потребителям:

– отправка новой версии ППО с сопроводительным письмом;

– публикация ISO-образа загрузочного модуля новой версии ППО на официальном веб-сайте предприятия-разработчика (https://www.omp.ru, https://auroraos.ru).

Потребитель также имеет возможность получить информацию о выходе обновлений через службу технической поддержки предприятия-разработчика по тел.: +7 (495) 269-09-80 либо по электронной почте: support@omp.ru.

Установка обновлений ППО должна осуществляться в соответствии с требованиями, приведенными в настоящем документе.

Если потребитель ППО не может реализовать компенсирующие меры по защите информации или ограничения по применению ППО, рекомендуется прекратить его применение.

ВНИМАНИЕ! При обновлении ППО допускается переход только на более новую версию. Даунгрейд (откат к предыдущей версии) не поддерживается в связи с необратимыми изменениями в структуре БД и схеме конфигурации, вносимыми при обновлении.

Обновление ППО до требуемой версии возможно только с версий, указанных в таблице (Таблица 33).

Таблица 33

Требуемая версия	Ранее установленная версия
2.2.0	2.1.3*
2.2.1*	2.1.3*, 2.2.0
2.2.2*	2.1.3*, 2.2.0, 2.2.1*
2.3.0	2.2.2*
2.4.0	2.2.2*, 2.3.0
2.5.0	2.2.2*, 2.3.0, 2.4.0
2.5.1*	2.2.2*, 2.3.0, 2.4.0, 2.5.0
3.0.0	2.5.1*
3.0.1	2.5.1*, 3.0.0
3.1.0	2.5.1*, 3.0.1
3.1.1*	2.5.1*, 3.0.1, 3.1.0
3.1.2*	2.5.1*, 3.1.0, 3.1.1*
3.2.0	3.1.0, 3.1.2*
4.0.0*	3.1.2*, 3.2.0
4.1.0*	3.1.2*, 4.0.0*
5.0.0*	3.2.0, 4.0.0*, 4.1.0*
5.1.0*	4.1.0*, 5.0.0*
5.2.0	4.1.0*, 5.0.0*, 5.1.0*
5.3.0*	4.1.0*, 5.0.0*, 5.1.0*, 5.2.0
5.4.0	5.0.0*, 5.1.0*, 5.2.0, 5.3.0*
5.4.1	5.0.0*, 5.1.0*, 5.2.0, 5.3.0*, 5.4.0
5.4.2	5.4.0, 5.4.1
5.4.3*	5.3.0*, 5.4.1, 5.4.2
5.5.0*	5.3.0*, 5.4.1, 5.4.2, 5.4.3*
5.6.0*	5.4.1, 5.4.2, 5.4.3*, 5.5.0*
5.6.1*	5.4.2, 5.4.3*, 5.5.0*, 5.6.0*
5.7.0*	5.4.3*, 5.5.0*, 5.6.0*, 5.6.1

* – версии ППО, прошедшие сертификацию в ФСТЭК России.

6.2. Обновление сервера приложений ППО

ВНИМАНИЕ! Для установки обновления ППО количество свободного места на жестком диске сервера БД ПБ должно быть не меньше, чем размер самой БД ПБ. При недостаточном количестве свободного места на жестком диске его необходимо увеличить. Продолжительность процесса обновления ППО зависит от размера БД и может занять длительное время.

Для обновления сервера приложений ППО необходимо выполнить описанные ниже действия:

6.2.1. Создать резервную копию данных, ППО и компонентов среды функционирования в соответствии с разделом 5.

6.2.2. Скопировать на управляющую ЭВМ архив с новой версией ППО и распаковать его в соответствии с п. 3.3.4 – 3.3.5.

6.2.3. Обновить на управляющей ЭВМ пакеты в соответствии с п. 3.3.7.

6.2.4. Настроить компоненты среды функционирования ППО и ППО в соответствии с подразделом 3.5.

ВНИМАНИЕ! Конфигурационные файлы в зависимости от версии ППО различаются, поэтому копирование их параметров осуществляется вручную.

6.2.5. Обновление СУБД PostgreSQL 14/15/16 до новой старшей версии[21] (`major version`) необходимо осуществлять в соответствии с ЭД на СУБД. Перечень поддерживаемых версий СУБД приведен в п. 1.4.2.

6.2.6. Установить компоненты среды функционирования ППО и инфраструктурные компоненты ППО в соответствии с п. 3.6.1.

6.2.7. Установить ППО в соответствии с п. 3.6.2.

6.2.8. Оповестить пользователей ППО о необходимости очистить кэш и cookies веб-браузера. Иначе при открытии интерфейса ППО будет ошибка HTTP ERROR 400.

[21] Согласно спецификации SemVer 2.0.0.

ПРИМЕЧАНИЕ. В случае, если в процессе обновления ППО возникает ошибка, связанная с nginx-пакетами, то рекомендуется выполнить следующие действия:

– удалить `consul_template` выполнив:

```
ANSIBLE_USER=omp ./deploy-infra.sh -c consul-template -A flush_all
```

– удалить `nginx` выполнив:

```
ANSIBLE_USER=omp ./deploy-infra.sh -c nginx -A flush_all
```

– очистить пакеты `nginx` с помощью команды:

```
sudo apt remove --purge nginx nginx-common
```

– повторно установить `nginx` выполнив:

```
ANSIBLE_USER=omp ./deploy-infra.sh -c nginx -A deploy
```

– затем установить компоненты ППО поочередно выполнив:

```
ANSIBLE_USER=omp ./deploy-infra.sh -A deploy
ANSIBLE_USER=omp ./deploy-ac.sh -A deploy
```

– проверить статусы `health-check` используя команду:

```
sudo /usr/bin/consul_service_health_check.sh | grep -v 200
```

ПРИМЕЧАНИЕ. После обновления сервера приложений ППО рекомендуется обновить мобильное приложение «Аврора Маркет» в соответствии с документом «Руководство пользователя. Часть 6. Приложение «Аврора Маркет» для операционной системы Аврора» и приложение «Аврора Центр» в соответствии с документом «Руководство пользователя. Часть 7. Приложение «Аврора Центр» для операционной системы Аврора».

6.3. Обновление и установка ППО при использовании единого docker-образа (All-in-One)

При обновлении и установке ППО при использовании единого docker-образа (All-in-One) необходимо:

1) Разархивировать образ:

```
tar -xf AC_5.7.0_docker.tar.gz
```

2) Загрузить образ в `docker`:

```
docker load --input AC_5.7.0_docker.tar
```

3) Если требуется сохранить данные между запусками контейнера ППО, то создать директорию для хранения данных:

```
mkdir -p postgresql
sudo chown -R 108:109 postgresql
```

4) Выбрать `AC_URI`, который будет использоваться для доступа к ППО. (в примере используется http://ac570.local:8009).

Если доменного имени `ac570.local` не существует, то его нужно прописать в `/etc/hosts`.

5) Запустить контейнер (HTTP):

```
docker run \
   --interactive \
   --tty \
   --privileged \
   --detach \
   --publish "8009:8009" \
   --publish "8025:8025" \
   --env "AC_URI=http://ac570.local:8009" \
   --name ac \
   --volume $(pwd)/postgresql:/var/lib/postgresql \
   ac:v5.7.0.55
```

6) Запустить контейнер с HTTPS/TLS. Для запуска контейнера с HTTPS требуется подготовить ключ и сертификат сервера, а также сертификат удостоверяющего центра:

– выпустить самоподписанный сертификат (опционально):

```
# Создание ключа и сертфката CA
openssl req -x509 -nodes -days 3650 -newkey rsa:4096 \
  -keyout CA.key \
  -out CA.crt \
  -subj "/C=RU/CN=My Local Root CA"
```

```
# Генерация ключа сервера и запроса (CSR)
openssl req -new -nodes -newkey rsa:2048 \
  -keyout ac570.local.key \
  -out ac570.local.csr \
  -subj "/C=RU/CN=ac570.local" \
  -addext "subjectAltName = DNS:ac570.local" \
  -addext "extendedKeyUsage = serverAuth"

# Выпуск сертификата
openssl x509 -req -days 365 \
  -in ac570.local.csr \
  -CA CA.crt \
  -CAkey CA.key \
  -CAcreateserial \
  -out ac570.local.crt \
  -copy_extensions copy

# Создание файла с цепочкой сертификатов
cat ac570.local.crt CA.crt > ac570.local.chain.crt
```

– запустить контейнер (HTTPS):

```
docker run \
   --interactive \
   --tty \
   --privileged \
   --detach \
   --publish "8009:8009" \
   --publish "8025:8025" \
   --env "AC_URI=http://ac570.local:8009" \
   --name ac \
   --volume $(pwd)/postgresql:/var/lib/postgresql \
   --volume "$(pwd)/ac570.local.key:/etc/consul-template/ssl.key" \
   --volume "$(pwd)/ac570.local.chain.crt:/etc/consul-
template/ssl.crt" \
   --volume "$(pwd)/CA.crt:/usr/local/share/ca-certificates/ca.crt" \
   ac:v5.7.0.55
```

7) Открыть консоль администратора:

– HTTP: http://ac570.local:8009/auth/admin/;

– HTTPS: https://ac570.local:8009/auth/admin/;

– admin@omprussia.ru/Admin123!

6.4. Обновление ОС Аврора с помощью ПУ

ВНИМАНИЕ! После обновления ППО рекомендуется выполнить настройки внешнего балансировщика в соответствии с п. 2.1.3.

Обновление ОС Аврора выполняется в следующей последовательности:

6.4.1. Обновить сервер приложений ППО в соответствии с подразделом 6.1.

6.4.2. Загрузить в файловое хранилище ПООС пакеты требуемой версии ОС в соответствии с п. 3.8.3.

6.4.3. Обновить ОС Аврора до требуемой версии на тестовой группе устройств с целью проверки корректности обновления с помощью политики «Приложения/Установить версию ОС». Порядок работы с политиками и группами устройств приведен в документе «Руководство пользователя. Часть 3. Подсистема Платформа управления».

6.4.4. Убедиться, что после окончания, заданного в правиле временного интервала обновления в карточке каждого устройства из тестовой группы отображается требуемая версия ОС Аврора.

6.4.5. Обновить приложения ППО на тестовой группе устройств в соответствии с документом «Руководство пользователя. Часть 7. Приложение «Аврора Центр» для операционной системы Аврора».

6.4.6. Выполнить обновление аналогичным образом для остальных устройств после успешного обновления ОС Аврора и приложений на тестовой группе устройств.

7. УДАЛЕНИЕ ППО

Для удаления ППО необходимо выполнить следующие действия:

7.1. Перейти в каталог со сценариями установки ППО.

7.2. Удалить ППО с помощью команды:

```
ANSIBLE_USER="<имя пользователя>" ./deploy-ac.sh --action flush_all
```

Подробное описание параметров запуска скрипта `deploy-ac.sh` приведено в подразделе 4.2.

7.3. Удалить компоненты среды функционирования ППО и инфраструктурные компоненты ППО с помощью команды:

– без удаления данных, хранящихся в СУБД PostgreSQL:

```
ANSIBLE_USER="<имя пользователя>" ./deploy-infra.sh --action flush_all
```

– с удалением данных, хранящихся в СУБД PostgreSQL:

```
ANSIBLE_USER="<имя пользователя>" ./deploy-infra.sh --action flush_all
--extra-vars "pg_uninstall_delete_data=true"
```

Подробное описание параметров запуска скрипта `deploy-infra.sh` приведено в подразделе 4.1.

8. ВАРИАНТЫ УСТАНОВКИ ПСУ

8.1. Установка ПСУ на один сервер (хост) с другими подсистемами ППО

Данный вариант установки ППО осуществляется по умолчанию.

8.2. Установка ПСУ на отдельный сервер (хост)

Задание адресов серверов (имен хостов), на которые будут установлены подсистемы ППО, осуществляется на этапе настройки ППО (пп. 3.5.2.1). Для того, чтобы установить ПСУ на отдельный сервер, необходимо в инвентарном файле `inventories/hosts.yml` задать адрес сервера (имя хоста), на который необходимо установить ПСУ (`push`), например:

```
...
        app:
          hosts:
            ocs-app.local:
              subsystems: auth,appstore,emm,mt,pkgrepo,
            acenterapp02:
              ocs-push.local: push
```

Описание порядка задания адресов в инвентарном файле `inventories/hosts.yml` приведено в п. 3.10.13.

8.3. Отдельная установка ПСУ (установка ПБ и ПСУ)

Для отдельной установки ПСУ необходимо выполнить следующую последовательность действий:

8.3.1. Выполнить последовательность действий, предусмотренную подразделом 3.2 – п. 3.6.1.

8.3.2. Установить ПБ и ПСУ с помощью команды:

```
ANSIBLE_USER="<имя пользователя>" ./deploy-ac.sh --subsystems
auth,push
```

8.3.3. Выполнить последовательность действий, предусмотренную п. 3.6.3 - 3.6.5.

9. ВАРИАНТЫ УСТАНОВКИ СУБД

9.1. Некластерная (standalone) установка СУБД

В данном варианте устанавливается только один сервер БД. Применяется такая установка в однонодовой конфигурации ППО (п 2.3.2 и 2.3.3).

Для того, чтобы выполнить `standalone` установку сервера БД достаточно в инвентарном файле `inventories/hosts.yml` в процессе настройки компонентов среды функционирования ППО и инфраструктурных компонентов ППО (п. 3.5.1) задать адрес хоста, на который будет установлен сервер БД, например:

```
...
        postgresql:
          hosts:
            ocs-db.local:
```

9.2. Установка СУБД в кластерной конфигурации

Кластерная установка сервера БД применяется в катастрофоустойчивой кластерной конфигурации ППО и предполагает установку кластеров СУБД в нескольких ЦОД (п. 2.3.7).

ВНИМАНИЕ! При установке ППО в кластерной конфигурации не допускается устанавливать серверы приложений ППО и серверы БД на одном хосте.

Для установки СУБД в кластерной конфигурации необходимо в процессе настройки компонентов среды функционирования ППО и инфраструктурных компонентов ППО (п. 3.5.1) выполнить следующие настройки:

9.2.1. В каталоге `inventories` со сценариями установки ППО создать окружения для основного (primary) и резервного (standby) кластеров с помощью следующих команд:

```
mkdir -p inventories/<название окружения>
cp inventories/hosts.yml inventories/<название окружения>/
```

Например:

```
mkdir -p inventories/ocs-primary
cp inventories/hosts.yml inventories/ocs-primary/
mkdir -p inventories/ocs-standby
cp inventories/hosts.yml inventories/ocs-standby/
```

Подробная информация о настройке ППО для установки его на различные окружения приведена в п. 3.10.16.

9.2.2. Выполнить настройку инвентарного файла `hosts.yml` для основного кластера.

Для этого в инвентарном файле `inventories/<название окружения>/hosts.yml` необходимо задать следующие параметры:

– `patroni_cluster` – название кластера. Название кластера должно иметь следующий формат `<название кластера>-<тип кластера>`. Тип кластера должен быть `primary` (основной). Например:

```
...
        patroni_cluster: ocs-primary
```

– `keepalived_cluster_vip` – виртуальный IP-адрес (`virtual IP`), который будет присвоен активному серверу БД в основном кластере, например:

```
...
        keepalived_cluster_vip:
          ocs-primary: 192.168.1.60
```

ПРИМЕЧАНИЕ. Виртуальный IP-адрес используется для подключения приложений (клиентов) к активному серверу БД.

– адреса хостов, на которые будут установлены серверы БД, например:

```
...
        postgresql:
          hosts:
            acenterdb01:
            acenterdb02:
```

– адреса хостов, на которые будут установлены агенты Consul. Т.к. агенты Consul устанавливаются на серверы БД, вместо хостов достаточно указать группу `postgresql`:

```
...
            consul_agents:
              children:
                postgresql:
```

ВНИМАНИЕ! Не допускается установка агента и сервера Consul на одном хосте.

Пример инвентарного файла для основного кластера приведен в файле `samples/ac/inventories/hosts-patroni-two-dbnode-primary.yml`.

9.2.3. Выполнить настройку инвентарного файла `hosts.yml` для резервного (standby) кластера.

Для этого в инвентарном файле `inventories/<название окружения>/hosts.yml` необходимо задать следующие параметры:

– `patroni_cluster` – название кластера. Название кластера должно иметь следующий формат `<название основного кластера>-<тип кластера>`. Тип кластера должен быть `standby` (резервный). Например:

```
...
        patroni_cluster: ocs-standby
```

– `keepalived_cluster_vip` - виртуальные IP-адреса (`virtual IP`) для активного сервера БД в основном кластере и для StandBy Leader сервера БД в резервном кластере, например:

```
...
        keepalived_cluster_vip:
          ocs-primary: 192.168.1.60
          ocs-standby: 192.168.1.61
```

– адреса хостов, на которые будут установлены серверы БД резервного кластера, например:

```
...
        postgresql:
          hosts:
            acenterdb11:
            acenterdb12:
```

– адреса хостов, на которые будут установлены агенты Consul. Т.к. агенты Consul устанавливаются на серверы БД, вместо хостов достаточно указать группу `postgresql:`

```
...
            consul_agents:
              children:
                postgresql:
```

ВНИМАНИЕ! Не допускается установка агента и сервера Consul на одном хосте.

Пример инвентарного файла для резервного кластера приведен в файле `samples/ac/inventories/hosts-patroni-two-dbnode-standby.yml`.

9.2.4. Выполнить другие настройки ППО и компонентов среды функционирования ППО в соответствии с подразделом 3.5.

9.2.5. Выполнить установку компонентов среды функционирования ППО и ППО в соответствии с подразделом 3.6 для каждого кластера, указав в командах установки путь к инвентарному файлу или название окружения.

Примеры команд:

– команда установки всех пакетов на серверы приложений и серверы БД:

```
ansible-playbook -i inventories/<название окружения>/hosts.yml play-managed-node-prerequisites.yml -vv -u <имя пользователя>
```

– команда установки компонентов среды функционирования ППО и инфраструктурных компонентов ППО:

```
ANSIBLE_USER="<имя пользователя>" ./deploy-infra.sh --env "<название окружения>"
```

– команда установки ППО:

```
ANSIBLE_USER="<имя пользователя>" ./deploy-ac.sh --env "<название окружения>"
```

10. УСТАНОВКА ППО В KUBERNETES

10.1. Порядок развертывания и настройки сервера приложений

В таблице (Таблица 34) приведены минимальные программно-аппаратные характеристики сервера приложений.

Таблица 34

Параметр	Значение	
	Testing	Production
Процессор	2 ядра	5 ядер
Объем оперативной памяти	8 Гб	12 Гб
Объем жесткого диска	60 Гб	60 Гб
ПО для управления контейнерами	Kubernetes версии 1.25 или выше, либо сборка от Canonical microk8s	

ПРИМЕЧАНИЕ. Kubernetes и Platform V DropApp являются взаимозаменяемыми компонентами.

ВНИМАНИЕ! Для повышения безопасности развертывания в Kubernetes допускается включение поддержки Platform V Synapse Service Mesh.

Реализуемые сценарии защиты трафика:

1) Управление межсервисным взаимодействием внутри Kubernetes. Platform V Synapse Service Mesh обеспечивает прозрачный `mTLS` (на основе сертификатов, управляемых Platform V Synapse Service Mesh) между сервисами внутри кластера. Трафик шифруется и аутентифицируется автоматически, что повышает общий уровень безопасности без изменения кода приложений;

2) Единая точка входа для внешнего трафика. Все входящие соединения проходят через `Istio Ingress Gateway`, который выполняет терминацию TLS, проверку клиентских сертификатов и маршрутизацию внешних запросов к `nginx-ingress`;

3) Контроль выхода сервисов кластера к внешним ресурсам (СУБД и сервис гарантированной доставки сообщений). Приложения обращаются к целевым внешним ресурсам без шифрования. Sidecar-контейнер Istio перехватывает соединение и через Istio Egress Gateway перенаправляет трафик к внешней СУБД или сервису гарантированной доставки сообщений (Redpanda/Kafka). На участке от `sidecar` до `egress`-шлюза действует автоматический mTLS с сертификатами, управляемыми Platform V Synapse Service Mesh. На участке от `egress`-шлюза до внешних ресурсов используется mTLS с явно заданными клиентскими сертификатами, CA и SNI. Это позволяет централизованно управлять исходящими подключениями, не разнося TLS-настройки по приложениям и не открывая прямой доступ из подов во внешнюю сеть.

Для включения функционала необходимо до развертывания ППО самостоятельно развернуть Platform V Synapse Service Mesh и настроить CRD SSM (на основе примеров из папки дистрибутива по пути `samples/synapse/`). Включение поддержки со стороны ППО реализуется путем установки переменной `istio_k8s_support: true` в файле `_vars.yml` среды.

Пример установки `microk8s` на ОС Ubuntu 24.04:

10.1.1. Установить на сервер приложений необходимые пакеты с помощью команды:

```
sudo apt update && sudo apt -y install snapd nfs-common
sudo apt install docker.io
sudo snap install microk8s –classic
```

ПРИМЕЧАНИЕ. Установка пакета nfs-common является обязательной для рабочих серверов, использующих сетевое хранилище по протоколу NFS. В случае применения альтернативных типов хранилищ установка данного пакета не требуется.

10.1.2. Установить требуемые системные компоненты. Для этого потребуется диск с клиентскими приложениями (более подробное описание приведено в подраздел 3.3), после чего распаковать и установить требуемые пакеты с помощью команд:

```
cd <путь к каталогу>
bash AC_<версия ППО>_installer-ac-mt.sh
cd install-<версия ППО>/install-ac-mt/
sudo ./cnp-system-pkgs.sh
./cnp-python-pkgs.sh
```

10.1.3. Включить расширение `hostpath-storage`, необходимое для сохранения данных `stateful`-приложений (приложений, которым требуется сохранять данные), при перезагрузке контейнеров с помощью команды:

```
sudo microk8s enable hostpath-storage
```

ПРИМЕЧАНИЕ. В случае отсутствия установленного сервера (`registry`) для docker images, потребуется осуществить его установку (данная возможность предназначена исключительно для ознакомительных целей). Для этого необходимо воспользоваться следующей командой:

```
sudo microk8s enable registry
```

10.1.4. Включить расширение `metallb`, необходимое для доступа к сервисам ППО (порты 80 и 8009) снаружи кластера, с помощью команды:

```
sudo microk8s enable metallb:<диапазон IP-адресов подсети сервера приложений>
```

Например:

```
sudo microk8s enable metallb:10.188.25.196-10.188.25.197
```

ПРИМЕЧАНИЕ. Число IP-адресов в диапазоне может быть не менее двух штук.

10.1.5. Убедиться, что расширения `microk8s`, `metallb` и `registry` успешно включены и находятся в списке доступных (`enabled`) расширений с помощью команды:

```
sudo microk8s status
```

10.1.6. Для удобства работы с Kubernetes рекомендуется установить консольный клиент `k9s`, а также `kubectl` с помощью команд:

```
sudo wget
https://github.com/derailed/k9s/releases/download/v0.32.3/k9s_linux_am
d64.deb
sudo dpkg -i k9s_linux_amd64.deb
sudo curl -LO https://dl.k8s.io/release/`curl -LS
https://dl.k8s.io/release/stable.txt`/bin/linux/amd64/kubectl
sudo chmod +x ./kubectl
sudo mv ./kubectl /usr/local/bin/kubectl
```

10.1.7. Для работы из-под непривилегированного пользователя необходимо дополнительно выполнить следующие команды:

```
USER=<имя пользователя>
sudo usermod -a -G microk8s $USER
sudo mkdir -p ~/.kube
sudo usermod -a -G docker $USER
sudo chown -f -R $USER ~/.kube
```

Для того, чтобы применились настройки из пп. 10.1.1 необходимо перезагрузить ОС или переподключиться по ssh к серверу, на котором осуществляется развертывание. В противном случае, описанную ниже команду придется выполнять под суперпользователем.

Далее выполните команду, которая предоставит доступ `kubectl`:

microk8s config > ~/.kube/config

10.2. Порядок установки ППО в Kubernetes

10.2.1. Перейти в каталог со сценариями установки `/install-<версия ППО>/install-ac-mt/`.

10.2.2. Для работы сценариев установки ППО необходимо установить в ansible-коллекции плагины `kubernetes.core` и `community.general` с помощью команды:

```
ansible-galaxy install --role-file requirements-k8s.yml --force
```

10.2.3. Сформировать конфигурационные файлы ППО.

Для этого необходимо придумать название окружения (например, `k8s`) и запустить скрипт генерации конфигурационных файлов, выполнив команду:

```
./deploy-k8s.sh -e <название окружения> -A configs
```

В процесс выполнения скрипта необходимо в интерактивном режиме задать следующие параметры:

– `deployment mode` – режим установки (`testing` или `production`). Режим `testing` рекомендуется использовать в ознакомительных целях. При использовании режима `testing`: все компоненты среды функционирования ППО и инфраструктурные компоненты ППО устанавливаются в одном экземпляре без резервирования, отсутствует необходимость в настройке внешнего балансировщика (все устанавливается на ingress-контроллере Kubernetes), контейнерам меньше выделяется ресурсов процессора и оперативной памяти;

– `external ip` – внешний IP-адрес, по которому будет доступно ППО (выбирается как любой свободный адрес из диапазона, который указывался при установке расширения `metallb`). Для просмотра заданного первоначально диапазона IP-адресов, необходимо воспользоваться командой:

```
kubectl -nmetallb-system get ipaddresspool -o
jsonpath="{.items[*].spec.addresses}"
```

– `DNS domain` – внешний домен, который вместе с названием окружения будет формировать URL-адрес ППО `http(s)://<название окружения>.<DNS domain>`;

– `cluster name` – имя кластера Kubernetes, в который будет установлено ППО. Кластер предварительно должен быть создан. Посмотреть список кластеров можно с помощью команды:

```
kubectl config get-clusters
```

– `storage class` – название класса хранилища Kubernetes (StorageClass), с которым будут создаваться тома хранения данных (volumes) для stateful-приложений. Посмотреть список классов хранилищ можно с помощью команды:

```
kubectl get storageclasses
```

– `k8s infra components[..]` - указывается перечень инфраструктурных компонентов, подлежащих установке в кластер Kubernetes, при этом для отказоустойчивых конфигураций инфраструктурные сервисы с высокими требованиями к дисковой подсистеме (например, СУБД PostgreSQL или брокер сообщений Redpanda) выносятся на выделенные серверы, и в таком случае в поле фиксируется список сервисов, разворачиваемых непосредственно в Kubernetes, а их внешняя настройка, с маппингом ролей на конкретные физические или виртуальные серверы, выполняется в соответствии с инструкциями по настройке `inventory`, начиная с подраздела 3.5;

– `public key` – открытый ssh-ключ пользователя ОС, из-под которого запускается установка ППО. При необходимости ключевую пару можно сформировать с помощью команды:

```
ssh-keygen -t rsa -b 4096 && cat ~/.ssh/id_rsa.pub
```

– `Docker registry basepath` – это адрес приватного `registry`, где будут сохранены `image`;

– `nfs server` – NFS сервер, используемый для хранения файлов приложений (иконки, скриншоты, RPM-пакеты) и пакетов ОС. Если предполагается использовать внешний сервер, то необходимо указать его IP-адрес (до установки на нем уже должны быть созданы каталоги согласно подразделу 3.9). Если готового NFS сервера нет, то необходимо оставить это поле пустым, в данном случае NFS сервер будет установлен автоматически внутри Kubernetes.

Пример задания параметров:

```
./deploy-k8s.sh -e k8s -A configs


Please answer a few questions for creating k8s stage configs and
inventories -

Choose Aurora Center deployment mode [testing/production]: testing

Aurora Center external IP: 10.188.25.196

Aurora Center external DNS domain: mydomain.ru

K8s cluster name: microk8s-cluster

K8s storage class name: microk8s-hostpath

SSH public key, that will be used for connections to the Management
Server:
ssh-rsa
AAAAB3NzaC1yc2EAAAADAQABAAABgQC5AOA8ANTqrVxIcCPrgkuXMXOIOy27Z629IKTM+D
xJTwjEj1n78a5oxr9P/rdGOSfNwFQRZ3pBeINCiYWB8OqvLmEOhCBQQ498EI64DnC7BepS
4tSQyiPnIaprChQBD9VBx4leMUMhoRl01tCHe0cqCBtjvRgp/3ZJScPLMCStU29cJ31OnP
P29+JduN9FRO5fl7/MG1h4MetWFcYc6D2eQJ78pAJ0+1COKEAzpzLRW4Qdw4mkI+ukOob2
uunEjOeows2yCumXcKiNpOcp0zV+Kt6y2o72rVWrPXN1Zs8wya7dmCULmjKzkOPl8GvA7f
fHByWXBBnvc9P7P/86wVwecLrjubmdNTAsD0NO+pd1vuuH89Pz8vQxjHEaNPho/Q38RhUX
5ElP4jsNIUVOsgVwdfnJPCdyL7KaI+GOd332YEB8tPHuUbCX+wOKCO10qzkwUPMFXTNh6v
HhO4OPthwbCEgZv5jgraW5D5JqcJXiEtSZZSMo9WUXaBsNbOtlGts= root@ubuntu22

Docker registry basepath for AC containers: 127.0.0.1:32000

NFS server for shared storage(ip/hostname or leave empty to deploy in-
cluster nfs server):
```

В результате выполнения скрипта, будут сформированы инвентарный файл окружения (файл: `inventories/<название окружения>/hosts.yml`), а также конфигурационные файлы окружения (каталог: `config/environments/<название окружения>/`). Подробная информация о конфигурационных файлах окружений приведена в п. 13.2.8. При необходимости изменения в данные файлы можно внести вручную.

ВНИМАНИЕ! Установка ППО предполагает наличие прав администратора (`cluster-admin`) в Kubernetes.

При отсутствии прав администратора необходимо выполнить следующие действия:

– сгенерировать минимальные права доступа (RBAC), необходимые для установки и обновления ППО с помощью команды:

```
./deploy-k8s.sh -e <название окружения> -A render_rbac
```

Например:

```
./deploy-k8s.sh -e k8s -A render_rbac
```

В результате выполнения скрипта, будут сформированы файлы с правами доступа для установки (файл: `_rendered/ocs-rbac/ocs-rbac-initial.yaml`) и обновления ППО (файл: `_rendered/ocs-rbac/ocs-rbac-update.yaml`).

– передать сгенерированные файлы администратору Kubernetes для создания учетной записи, с которой будет произведена установка или обновление ППО.

10.2.4. Создать docker-образы сервисов ППО.

10.2.4.1. Задать базовую ОС:

Для этого в конфигурационном файле `config/environments/<название окружения>/vars/_vars.yml` необходимо задать следующие параметры:

```
docker_os: <OS> # ubuntu|sberlinux|astralinux|redos|altlinux
docker_registry_basepath: <docker_registry_basepath>
```

Например:

```
docker_os: ubuntu # ubuntu|sberlinux|astralinux|redos|altlinux
docker_registry_basepath: "docker.test.ru/ac"
```

ПРИМЕЧАНИЕ. Чтобы обновить версии базовых образов ОС необходимо добавить в файл `config/vars/_vars.yml` переменную `docker_initial_images`.

Например:

```
docker_initial_images:
  ubuntu: ubuntu:22.04
  sberlinux: sberlinux.sbertech.ru/ubi9/ubi
  redos: registry.red-soft.ru/redos7c/ubi:latest
```

```
  altlinux: registry.altlinux.org/alt/alt:c10f2
  astralinux: registry.astralinux.ru/library/astra/ubi18:latest
  nginx: nginx:1.31.6
  nfs-server: phusion/baseimage:noble-1.0.0
```

10.2.4.2. Указать тип установки

Есть 2 варианта развертывания ППО в Kubernetes:

ВНИМАНИЕ! Если требуется перейти с второго варианта развертывания ППО на первый, необходимо переустановить ППО. Для этого необходимо вначале удалить ППО, не затрагивая инфраструктурные компоненты, следуя п. 10.3, затем развернуть ППО, следуя инструкции ниже.

1) Со сборкой отдельного docker-образа для каждого сервиса ППО;

2) Со сборкой общего docker-образа, используемого для запуска контейнеров ППО, в которые исполняемые модули будут загружены в момент запуска.

Для этого в конфигурационном файле `config/environments/<название окружения>/vars/_vars.yml` необходимо задать следующий параметр:

– варианта 1:

```
k8s_per_service_images: true
```

– варианта 2:

```
k8s_per_service_images: false
```

10.2.4.3. Залогиниться в `docker`-реестр, куда будут загружены собираемые образы. В случае, если используется плагин `registry y microk8s`, то это пункт можно пропустить.

Для этого необходимо выполнить следующую команду:

```
docker login <registry>
```

Например:

```
docker login docker.test.ru
```

10.2.4.4. Выполнить сборку контейнеров и переместить их в `docker-registry` с помощью команды:

```
./deploy-k8s.sh -e <название окружения> -A build_docker_images
```

Например:

```
./deploy-k8s.sh -e k8s -A build_docker_images
```

10.2.5. Добавить инфраструктурные образы ППО. Для этого требуется получить отдельный архив с инфраструктурными образами.

10.2.5.1. Распаковать архив с инфраструктурными образами ППО с помощью команды:

```
tar -xzf ./docker-images.tar.gz
```

10.2.5.2. Загрузить распакованные образы из директории `docker-images` в `docker`-реестр с помощью команды:

```
docker load -i docker-images/<инфраструктурный образ>
```

Например:

```
for item in $(ls docker-images/); do docker load -i docker-
images/$item; done
Loaded image: nginx/nginx-ingress:5.6.3
Loaded image: nginx:1.31.6
Loaded image: otel/opentelemetry-collector-contrib:0.155.0
Loaded image: docker.redpanda.com/redpandadata/redpanda:v26.2.2
Loaded image: valkey/valkey:9.1.2
```

10.2.5.3. Добавить теги на образы в соответствии с репозиторием, куда они будут загружены, например:

```
docker tag nginx:1.31.6 127.0.0.1:32000/nginx:1.31.6
docker tag nginx/nginx-ingress:5.6.3 127.0.0.1:32000/nginx/nginx-
ingress:5.6.3
docker tag valkey/valkey:9.1.2 127.0.0.1:32000/valkey/valkey:9.1.2
docker tag docker.redpanda.com/redpandadata/redpanda:v26.2.2
127.0.0.1:32000/redpandadata/redpanda:v26.2.2
docker tag otel/opentelemetry-collector-contrib:0.155.0
127.0.0.1:32000/otel/opentelemetry-collector-contrib:0.155.0
```

10.2.5.4. Загрузить образы в свой репозиторий, например:

```
docker push 127.0.0.1:32000/nginx:1.31.6
docker push 127.0.0.1:32000/nginx/nginx-ingress:5.6.3
```

```
docker push 127.0.0.1:32000/valkey/valkey:9.1.2
docker push 127.0.0.1:32000/redpandadata/redpanda:v26.2.2
docker push 127.0.0.1:32000/otel/opentelemetry-collector-
contrib:0.155.0
```

10.2.5.5. Указать образы в блоке `k8s_image_names` конфигурационного файла `config/environments/<название окружения>/vars/_vars.yml`, например, изменив `config/environments/k8s/vars/_vars.yml`:

```
k8s_image_names:
  app: "{{ docker_registry_basepath }}/ocs-base-image:{{
docker_ac_images['base_image']['version'] }}-{{ docker_os }}"
  init: "{{ docker_registry_basepath }}/ocs-base-image:{{
docker_ac_images['base_image']['version'] }}-{{ docker_os }}"
  ssh-server: "{{ docker_registry_basepath }}/ocs-ssh-server:{{
docker_ac_images['ssh_server']['version'] }}-{{ docker_os }}"
  nfs-server: "{{ docker_registry_basepath }}/ocs-nfs-server:{{
docker_ac_images['nfs_server']['version'] }}"

  nginx: "{{ docker_registry_basepath }}/nginx:1.31.6"
  nginx-ingress: "{{ docker_registry_basepath }}/nginx/nginx-
ingress:5.6.3"
  valkey: "{{ docker_registry_basepath }}/valkey/valkey:9.1.2"
  redpanda: "{{ docker_registry_basepath
}}/redpandadata/redpanda:v26.2.2"
  collector: "{{ docker_registry_basepath }}/otel/opentelemetry-
collector-contrib:0.155.0"
```

10.2.6. Выполнить установку ППО с помощью команды:

```
./deploy-k8s.sh -e <название окружения>
```

Например:

```
./deploy-k8s.sh -e k8s
```

ПРИМЕЧАНИЕ. В процессе установки ППО, также будут запущены образы инфраструктурных компонентов, которые были запушены ранее в `registry` – `nginx-ingress`, `nginx`, `redpanda`, `valkey`, `opentelemetry-collector-contrib`.

В случае успешной установки будет выведено сообщение следующего вида:

```
Congratulations, all done!

You can access AC from the following URLs -
  UI:                http://k8s.mydomain.ru/auth/admin/
```

```
  management-server: ssh ocs@k8s.mydomain.ru

!Warning!
check that k8s.mydomain.ru has a correct DNS record
  k8s.mydomain.ru A 10.188.25.196
or alternatively add to your local /etc/hosts file
  10.188.25.196 k8s.mydomain.ru
```

Первоначальный вход в ППО осуществляется с помощью Консоли администратора ПБ и предустановленной учетной записи в соответствии с подразделом 3.7.

10.3. Порядок удаления ППО из Kubernetes

Удаление ППО из Kubernetes осуществляется с помощью следующих команд:

```
bash deploy-k8s.sh -e <название окружения> -A delete
kubectl -n <название окружения> delete pvc --all
kubectl delete ns <название окружения>
```

Например:

```
bash deploy-k8s.sh -e k8s -A delete
kubectl -n k8s delete pvc --all
kubectl delete ns k8s
```

Если требуется удалить только ППО не затрагивая инфраструктурные компоненты (Valkey, Red Panda и т.д.), необходимо выполнить следующую команду:

```
bash deploy-k8s-ac.sh -e <название окружения> -A delete
```

Например:

```
bash deploy-k8s.sh -e k8s -A delete
```

11. ИМПОРТ ОТЧЕТОВ И НАСТРОЙКА АВТОМАТИЧЕСКОЙ РАССЫЛКИ В GRAFANA

11.1. Порядок импорта отчетов в Grafana

ВНИМАНИЕ! Функции импорта и автоматической рассылки отчетов не входили в объем сертификационных испытаний.

ПРИМЕЧАНИЕ. Тестирование проводилось для Grafana версии 11.6.2.

Для импорта отчетов в Grafana необходимо выполнить следующие действия:

11.1.1. Установить Grafana в соответствии с инструкцией с сайта разработчика (https://grafana.com/grafana/download?pg=get&platform=docker&plcmt=selfmanaged-box1-cta1&edition=oss) выполнив команду:

```
docker run -d --name=grafana -p 3000:3000 grafana/grafana:11.6.2
```

ПРИМЕЧАНИЕ. Веб-интерфейс Grafana в данном случае будет доступен по адресу `http://localhost:3000/` (логин: `admin`, пароль: `admin`).

ПРИМЕЧАНИЕ. При использовании плагина для рассылки отчетов из Grafana на почтовые адреса необходимо воспользоваться инструкцией в соответствии с пп. 0

11.1.2. Создать пользователя и выдать ему права на чтение, запись, обновление и удаление данных из таблицы в схеме `grafana_reports` с помощью скрипта `create_reporting_user.sql`.

11.1.2.1. При использовании плагина для рассылок отчетов из Grafana на почтовые адреса рекомендуется сначала создать схему `grafana_reports` в БД `emm` в соответствии с пп. 11.2.3.2, а затем выполнить скрипт `create_reporting_user.sql`.

11.1.2.1.1 Для выполения команд необходим доступ к PostgreSQL с правами администратора (суперпользователя). Можно использовать терминальную утилиту `psql` или любой графический SQL-клиент.

11.1.2.1.2 Подключиться к системной БД (по умолчанию `postgres`). Для подключения через терминал на сервере БД необходимо использовать команду:

```
psql –U postgres –d postgres
```

11.1.2.1.3 Создать пользователя и разрешить подключение к целевым базам. Выполнить следующие SQL-команды.

```
CREATE ROLE bi WITH LOGIN PASSWORD 'CHANGEME_Hefe7$';
GRANT CONNECT ON DATABASE emm TO bi;
GRANT CONNECT ON DATABASE auth TO bi;
```

ПРИМЕЧАНИЕ. Заменить `CHANGEME_Hefe7$` на надежный пароль.

11.1.2.1.4 После создания пользователя подключиться к БД `emm`. Если используется `psql`, необходимо переключить на нужную БД с помощью команды:

```
/c emm
```

ПРИМЕЧАНИЕ. При повторном использовании терминала необходимо выполнить команду для входа:

```
psql –U postgres –d emm
```

11.1.2.1.5 Настроить права в БД emm. Выполнить следующий блок SQL-команд для выдачи прав на схемы и таблицы:

– доступ к схемам:

```
GRANT USAGE ON SCHEMA device_inventory TO bi;
GRANT USAGE ON SCHEMA device_state_manager TO bi;
GRANT USAGE ON SCHEMA users_service TO bi;
GRANT USAGE ON SCHEMA applications TO bi;
GRANT USAGE ON SCHEMA policy TO bi;
```

– доступ к таблицам:

```
GRANT SELECT ON ALL TABLES IN SCHEMA device_inventory TO bi;
GRANT SELECT ON ALL TABLES IN SCHEMA device_state_manager TO bi;
GRANT SELECT ON ALL TABLES IN SCHEMA users_service TO bi;
GRANT SELECT ON ALL TABLES IN SCHEMA applications TO bi;
GRANT SELECT ON ALL TABLES IN SCHEMA policy TO bi
```

– расширенные права для схемы `grafana_reports`:

```
GRANT USAGE, CREATE ON SCHEMA grafana_reports TO bi;
GRANT USAGE, SELECT ON ALL SEQUENCES IN SCHEMA grafana_reports TO bi;
GRANT SELECT, INSERT, UPDATE, DELETE ON ALL TABLES IN SCHEMA
grafana_reports TO bi;
```

11.1.2.1.6 Подключиться к БД `auth`. В консоли `psql` переключиться на БД авторизации:

```
/c auth
```

ПРИМЕЧАНИЕ. Для подключения через терминал использовать команду:

```
psql –U postgres –d auth.
```

11.1.2.1.7 Выполнить завершающий блок SQL-команд:

– доступ к схемам:

```
GRANT USAGE ON SCHEMA audit TO bi;
GRANT USAGE ON SCHEMA accounts_users TO bi;
```

– доступ к таблицам:

```
GRANT SELECT ON ALL IN SCHEMA audit TO bi;
GRANT SELECT ON ALL TABLES IN SCHEMA accounts_users TO bi;
```

11.1.2.2. При отсутствии плагина необходимо выполнить скрипт `create_reporting_user.sql` после удаления из него следующих команд:

```
-- Grant access schemas emm
GRANT USAGE, SELECT ON ALL SEQUENCES IN SCHEMA grafana_reports TO bi;
-- Grant access to tables emm
GRANT USAGE, CREATE ON SCHEMA grafana_reports TO bi;
GRANT SELECT, INSERT, UPDATE, DELETE ON ALL TABLES IN SCHEMA
grafana_reports TO bi;
```

11.1.3. Подключить БД ПУ (`emm`), ПБ (`auth`) и ПМ (`appstore`) в качестве источника данных (Data Source). Для этого необходимо выполнить следующие действия:

11.1.3.1. Перейти на форму выбора источника данных (`menu -> Connections -> Data sources`).

11.1.3.2. Нажать кнопку «Add data source».

11.1.3.3. Выбрать СУБД PostgreSQL.

11.1.3.4. Заполнить следующие параметры подключения к БД `emm`:

– Name: название источника данных (произвольное название, например, AC_emm);

– Host URL: адрес и порт СУБД;

– Database name: emm;

– Username: имя пользователя СУБД, созданного при помощи скрипта create_reporting_user.sql;

– Password: пароль пользователя СУБД, созданный при помощи скрипта create_reporting_user.sql;

– TLS/SSL Mode: disabled;

– Version: версия СУБД PostgreSQL или PostgresPro, используемая в ППО.

11.1.3.5. Нажать кнопку «Save and Test».

11.1.3.6. Повторить пункты 11.1.3.1-11.1.3.5, используя учетные данные БД `auth` и `appstore` для подключения.

11.1.4. Выполнить импорт отчетов, которые находятся в каталоге `samples/dashboards/`.

Для импорта отчетов необходимо выполнить следующие действия:

11.1.4.1. Перейти к форме импорта отчетов (`menu -> Dashboards`).

11.1.4.2. Нажать кнопку «New».

11.1.4.3. Из раскрывающегося списка выбрать «Import».

11.1.4.4. Добавить отчет `devices-dashboard.json` в область загрузки JSON-файла.

11.1.4.5. Нажать кнопку «Import».

В результате в Grafana появится информационная панель (`dashboard`) с отчетом. Пример приведен на рисунке (Рисунок 43).



Рисунок 43

11.1.5. Повторить пп. 11.1.4.1-11.1.4.5 для других отчетов.

ПРИМЕЧАНИЕ. В отчеты «Список приложений не под управлением по устройствам» и «Список приложений не под управлением» не попадают списки системных приложений ОС Android, ОС Аврора, РЕД ОС и ОС Альт Linux.

11.2. Установка и настройка плагина для автоматической рассылки отчетов из Grafana на почту

ПРИМЕЧАНИЕ. Плагин протестирован на Grafana версии 11.6.2

11.2.1. Описание плагина «OMP Grafana Email»

ПРИМЕЧАНИЕ. Данный плагин не относится к сертифицированным дистрибутивам ППО.

Плагин «OMP Grafana Email» позволяет создавать и настраивать автоматические рассылки отчетов списку адресатов. При создании или настройке расписания можно выбрать нужный дашборд, применить фильтры для данных, задать список получателей и периодичность формирования отчета и его рассылки.

ПРИМЕЧАНИЕ. Для корректной работы с просмотром или настройкой дашборда с рассылками плагина – необходимо на стороне пользовательской ЭВМ выдать сетевой доступ до развернутой серверной части плагина.

В результате осуществления рассылки в соответствии с заданным расписанием адресатам автоматически приходит сформированный файл в формате XLSX, содержащий таблицы и диаграммы из дашборда. Полученный файл пригоден для визуального анализа и последующей обработки как вручную, так и средствами автоматизации.

ПРИМЕЧАНИЕ. Сформированный файл для рассылки может иметь достаточно большой размер и почтовый сервер может не пропустить его, поэтому рекомендуется настроить почтовый сервер на отправку писем с вложениями до 150-200 МБ.

11.2.2. Состав плагина OMP Grafana Email

Плагин состоит из двух компонентов:

– Компонент «OMP Grafana Email Panel Plugin» обеспечивает графический интерфейс для настройки и управления рассылок и их расписаниями непосредственно в панели Grafana. В него входит отчет с помощью которого осуществляется управление рассылками и их расписаниями `dashboard.json`, находящийся в архиве файла `provisioning` по пути `/provisioning/dashboards`.

– Компонент «OMP Grafana Email Backend» реализует серверную логику плагина, обрабатывая запросы со стороны интерфейса и выполняя необходимые операции на стороне сервера.

11.2.3. Предварительные шаги до установки и настройки плагина

11.2.3.1. Создание технической учетной записи и токена для нее

11.2.3.1.1 Технические учетные записи и их токены могут быть использованы для аутентификации через API Grafana. Для того, чтобы создать техническую учетную запись необходимо перейти в «Administration» → «Users and access» → «Service accounts» (Рисунок 44).

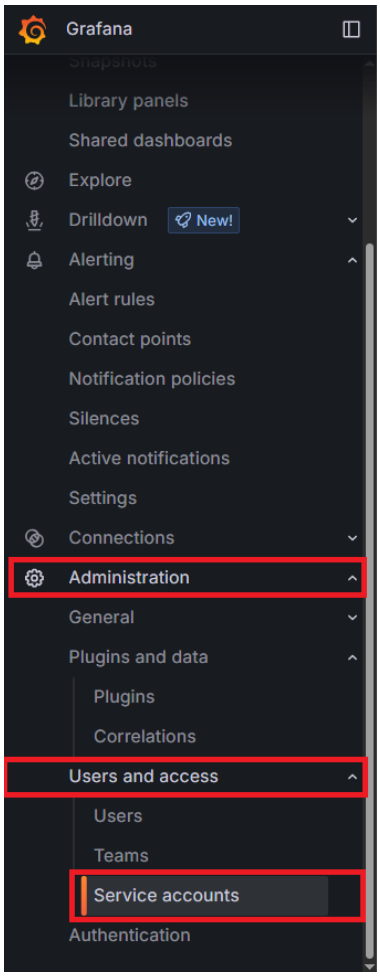


Рисунок 44

11.2.3.1.2 Нажать на кнопку «Add service account» (Рисунок 45).

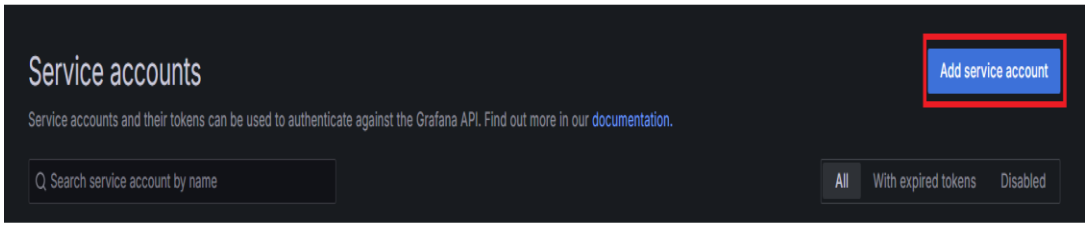


Рисунок 45

11.2.3.1.3 Создать техническую учетную запись, выполнив следующие действия:

1) Задать имя учетной записи в поле Display name (Рисунок 46);

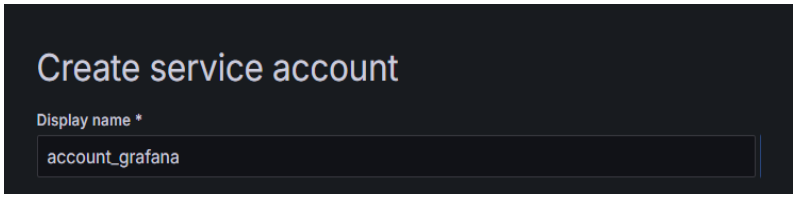


Рисунок 46

2) Указать роль Viewer в поле «Role» (Рисунок 47);

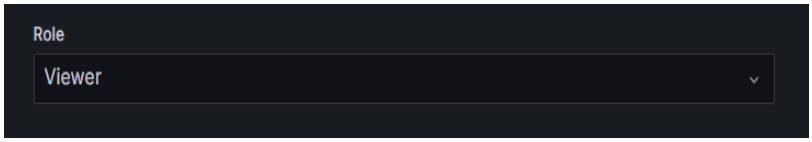


Рисунок 47

3) Нажать кнопку «Create» (Рисунок 48).

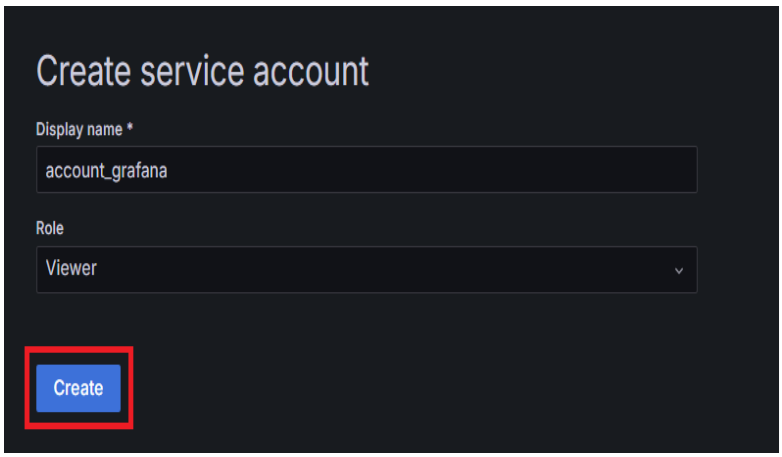


Рисунок 48

11.2.3.1.4 Сгенерировать токен для технической учетной записи. Для этого нажать на кнопку «Add service account token» (Рисунок 49).

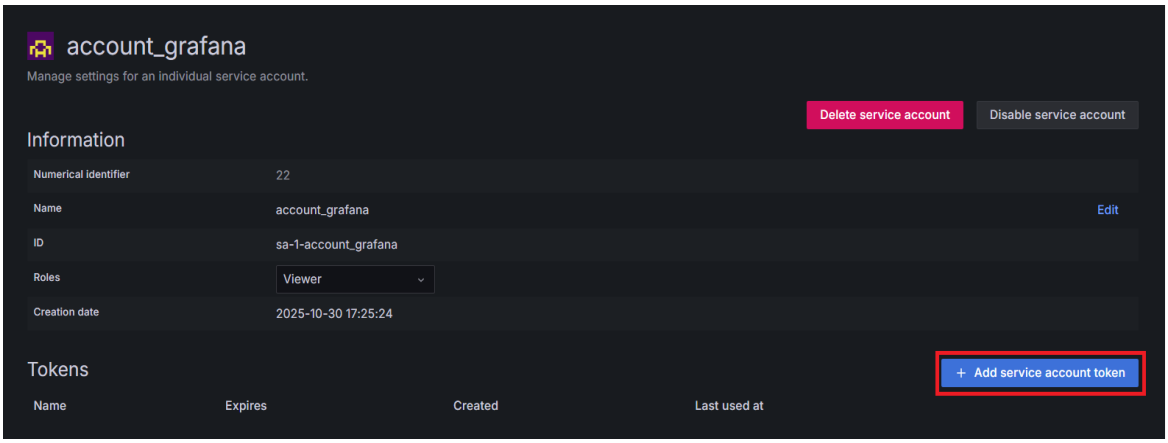


Рисунок 49

11.2.3.1.5 В открывшемся модальном окне «Add service account token» при необходимости задать срок жизни токена.

ВНИМАНИЕ! По истечении срока жизни токена его необходимо будет сгенерировать заново (только для токенов со сроком жизни). Для этого необходимо изменить значение параметра `grafanaAPIKey` в конфигурационном файле `config.json` и перезапустить сервис с помощью команды:

```
systemctl restart grafana-sender
```

11.2.3.1.6 Нажать кнопку «Generate token» (Рисунок 50).

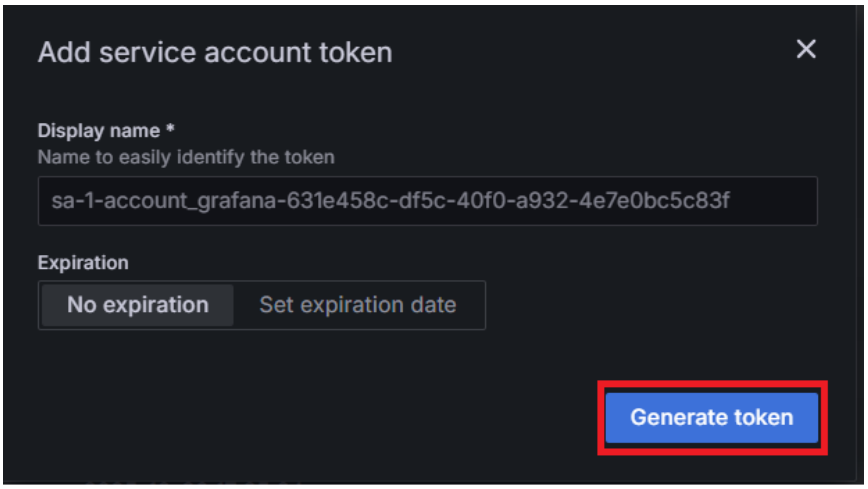


Рисунок 50

11.2.3.1.7 В открывшемся модальном окне «Service account token created» скопировать и сохранить сгенерированный токен, после чего нажать кнопку «Copy to clipboard and close» (Рисунок 51).

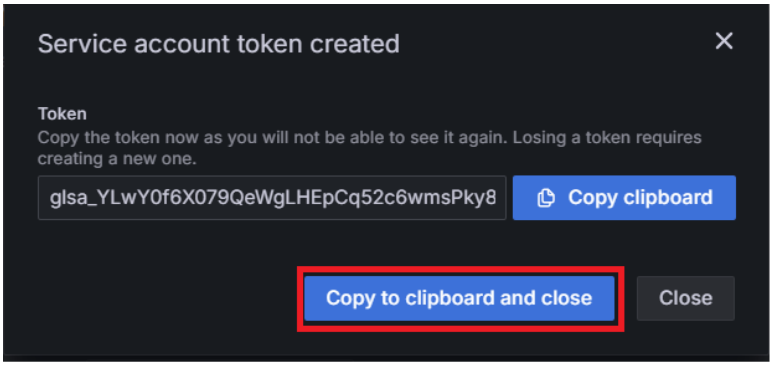


Рисунок 51

11.2.3.2. Добавление схемы в БД для настроек расписания плагина

Необходимо создать новую схему для подключенной БД ППО с целью сохранения настроек расписания рассылок и их корректной работы с плагином.

ПРИМЕЧАНИЕ. Рекомендуется создать схему в БД emm.

ВНИМАНИЕ! Указанные в данном пункте методы являются примерами того, как можно создать новую схему в БД. Допустимо использование другого способа.

1) Через терминал:

– подключиться с правами суперпользователя к БД, для которой требуется создать схему;

– создать схему с помощью команды:

```
CREATE SCHEMA grafana_reports;
```

– создать пользователя с правами на запись/чтение из созданной схемы `grafana_reports`, выполнив sql-скрипт `create_reporting_user.sql`.

2) Через инструмент управления БД (на примере pgAdmin):

– найти БД, в которой необходимо создать схему (рекомендуется emm);

– нажать правой кнопкой мыши на БД и выбрать Create (Создать) -> Schema (Схема);

– в открывшемся окне перейти на вкладку «General» (Общие) и ввести имя для новой схемы в поле «Schema name» (Имя схемы);

– нажать кнопку «Save» (Сохранить);

– создать пользователя с правами на запись/чтение из созданной схемы `grafana_reports`, выполнив `sql`-скрипт `create_reporting_user.sql`.

11.2.3.3. Создание почтового ящика на SMTP-сервере

ПРИМЕЧАНИЕ. Если есть созданный почтовый ящик на SMTP-сервере, который требуется в дальнейшем использовать для рассылки при помощи аутентификации на SMTP-сервере, то данный шаг можно пропустить.

Для осуществления рассылки можно использовать анонимный почтовый адрес или создать почтовый ящик на своем SMTP-сервере, для этого необходимо обратиться к системному администратору SMTP-сервера.

11.2.4. Настройка конфигурации плагина перед установкой

В конфигурационном файле `config.json` необходимо заполнить следующие параметры для настройки:

– `grafanaBaseUrl`: URL Grafana;

– `grafanaAPIKey`: токен, сгенерированный в пп. 11.2.3.1;

– `mail`:

- `defaultSubject`: тема письма для рассылки;
- `emailFrom`: адрес электронной почты - анонимный или созданный на SMTP-сервере;
- `emailUserName`: логин для идентификации на почтовом сервере (опциональный параметр, при отсутствии рассылка будет осуществляться с анонимного почтового ящика);
- `emailPassword`: пароль для аутентификации на почтовом сервере (опциональный параметр, при отсутствии рассылка будет осуществляться с анонимного почтового ящика);
- `smtpHost`: адрес до SMTP-сервера;
- `smtpPort`: порт SMTP-сервера;
- `useTLS`: использование TLS;

– `dbUser`: логин пользователя, созданного при помощи скрипта `create_reporting_user.sql`;

– `dbPassword`: пароль пользователя, созданный при помощи скрипта `create_reporting_user.sql`;

– `dbHost`: указывается адрес хоста БД;

– `database`: указывается имя БД;

– `searchPath`: схема, созданная в пп. 11.2.3.2;

– `httpPort`: указывается порт, на котором запущена серверная часть плагина.

Например:

```
{
    "grafanaBaseUrl": "http://localhost:3001",
    "grafanaAPIKey": "glsa_TZlyDJn3Si70G9ejTuiyFQ16ScOzcTqK_b558286f",
    "mail": {
        "defaultSubject": "Grafana Report",
        "emailFrom": "example@gmail.com",
        "emailUserName": "example@gmail.com",
        "emailPassword": "password",
        "smtpHost": "smtp.gmail.com",
        "smtpPort": 465 ,
        "useTLS": true
    },
    "debugModel":true,
    "pdfTitleAlign":"left",
    "fontPath":"",
    "dbUser":"bi",
    "dbPassword":"bi",
    "dbHost":"localhost:5432",
    "database":"emm",
    "searchPath":"grafana_reports",
    "httpPort": 10511,
    "cors": {
        "allowOrigin": "*",
        "allowCredentials": "true",
        "allowHeaders": "*",
        "allowMethods": "*"
    }
}
```

11.2.5. Установка плагинов для Grafana

11.2.5.1. Загрузка компонента OMP Grafana Email Panel Plugin

11.2.5.1.1 Для локально установленной Grafana

ВНИМАНИЕ! Директории из примера могут отличатся от ваших директорий. Также необходимо установить плагин для генерации изображений в отчетах, которые будут отправляться на почтовые адреса. Установка плагина Grafana-image осуществляется в соответствии с официальной документацией Grafana (https://grafana.com/docs/grafana/latest/setup-grafana/image-rendering/).

Директории указаны в конфигурационном файле `grafana.ini`, например:

```
. . .
# Directory where grafana will automatically scan and look for plugins
;plugins = path/plugins

# folder that contains provisioning config files that grafana will
apply on startup and while running.
;provisioning = path/provisioning
. . .
```

Для загрузки компонента необходимо:

1) Удалить символ «;» и в переменной `path` указать путь до папок `plugins` и `provisioning` в конфигурационном файле `grafana.ini`;

2) В конфигурационном файле `grafana.ini` задать значение параметра `allow_loading_unsigned_plugins` удалив символ «;»:

```
      . . .
      # Enter a comma-separated list of plugin identifiers to identify
plugins to load even if they are unsigned. Plugins with modified s>
      allow_loading_unsigned_plugins = omp-grafanaemail-panel
      . . .
```

3) Создать папку `omp-grafanaemail-panel` в директории `path/plugins/`;

4) Скопировать содержимое архива `samples/grafana/plugins/omp-grafanaemail-panel-plugin.tar.gz` в `path/plugins/omp-grafanaemail-panel/`, где `path` – путь до папки `omp-grafanaemail-panel`.

5) Скопировать содержимое архива `samples/grafana/plugins/omp-grafanaemail-panel-provisioning.tar.gz` в `path/provisioning/`, где `path` – путь до папки `provisioning`.

6) В файле `path/provisioning/dashboards/default.yml` в параметре `providers.options.path` необходимо указать путь до дашборда с рассылкой, например:

```
apiVersion: 1

providers:
  - name: 'Grafana Email'
    type: file
    allowUiUpdates: true
```

```
    options:
      path: path/provisioning/dashboards
```

7) Перезапустить сервис с помощью команды:

```
systemctl restart grafana-server
```

11.2.5.1.2 Для Grafana, собранной через Docker

ПРИМЕЧАНИЕ. Указанный метод является примером загрузки и запуска компонента серверной части. Допустимо использование другого способа. Установка плагина Grafana-image осуществляется в соответствии с официальной документацией Grafana (https://grafana.com/docs/grafana/latest/setup-grafana/image-rendering/).

Для загрузки компонента необходимо:

– создать файл `docker-compose.yaml` со следующим содержимым:

```
services:
  grafana:
    user: root
    network_mode: host
    image: grafana/grafana:11.6.2
    container_name: test-grafana
    restart: unless-stopped
    ports:
      - 3000:3000
    environment:
      GF_PLUGINS_ALLOW_LOADING_UNSIGNED_PLUGINS: omp-
grafanaemail-panel
      GF_RENDERING_SERVER_URL: http://localhost:8084/render
      GF_RENDERING_CALLBACK_URL: http://localhost:3000/
    volumes:
      - grafana-storage:/var/lib/grafana
      - path_on_local/provisioning:path_on_docker/provisioning
      - path_on_local/dist:path_on_docker/plugins/omp-
grafanaemail-panel
  renderer:
    image: grafana/grafana-image-renderer:latest
    network_mode: host
    ports:
      - 8084
volumes:
  grafana-storage: {}
```

– выполнить команду:

```
docker compose up
```

11.2.5.2. Загрузка компонента OMP Grafana Email Backend

ПРИМЕЧАНИЕ. Указанный метод является примером загрузки и запуска компонента серверной части. Допустимо использование другого способа.

ВНИМАНИЕ! Файл `config.json` должен быть предварительно настроен, иначе необходимо выполнить команду для перезапуска сервиса. Предварительная настройка плагина перед установкой описана в пп. 11.2.4.

Для загрузки компонента необходимо:

– получить из архива `samples/grafana/grafana-email/grafana-email.tar.gz` исполняемый файл `grafana-email` и создать файл `/etc/systemd/system/grafana-sender.service` со следующим содержимым:

```
[Unit]
Description=Grafana Sender service

[Service]
Restart=always
User=root
Group=root
ExecStart=path_to_binary/grafana-email \
    --config=path_config/config.json
WorkingDirectory=/tmp/
ExecStop=/usr/bin/kill grafana-email

[Install]
WantedBy=multi-user.target
```

– вместо переменной `path_to_binary` указать путь, ведущий до исполняемого файла `grafana-email`;

– в `--config` указать абсолютный путь до файла `config.json`, который был настроен в пп. 11.2.4;

– перезапустить демон `system` с помощью команды:

```
systemctl daemon-reload
```

– запустить сервис плагина с помощью команды:

```
systemctl start grafana-sender
```

– включить автозапуск сервиса плагина:

```
systemctl enable grafana-sender
```

11.3. Импорт дашборда в Grafana с использованием Prometheus

11.3.1. Подготовка исходных данных

Исходный `endpoint` для получения метрик:

```
http://<server>:8081/metrics
```

Например:

```
http://ac-app-server:8081/metrics
```

11.3.2. Установка и настройка Prometheus

11.3.2.1. Установить Prometheus

1) Скачать последнюю версию с официального сайта (https://prometheus.io);

2) Распоковать архив в целевую директорию;

3) Настроить конфигурационный файл `prometheus.yml`.

11.3.2.2. Настроить конфигурацию

Внести следующие изменения в конфигурационный файл `prometheus.yml`:

```
scrape_configs:
  - job_name: 'ac_metrics'
    static_configs:
      - targets:
        - ac-app-server:8081  # Добавьте все необходимые серверы
```

11.3.3. Настроить сбор метрик

1) Добавить все серверы приложений ППО в конфигурацию Prometheus;

2) Проверить доступность эндпоинта с помощью команды:

```
curl http://<server>:8081/metrics
```

Например:

```
curl http://ac-app-server:8081/metrics
```

3) Перезапустить Prometheus для применения изменений

11.3.4. Установить Grafana

1) Выполнить установку в соответствии с пп. 11.1.1;

2) Запустить сервис Grafana.

11.3.5. Настроить источник данных

1) Войти в интерфейс Grafana;

2) Перейти в «Configuration» - «Data Sources»;

3) Добавить новый источник данных:

– Тип: «Prometheus»

– URL: Адрес вашего сервера Prometheus

– Сохранить настройки

11.3.6. Импортировать дашборд

1) Перейти в раздел «Dashboards»;

2) Нажать кнопку «Import»;

3) Выбрать способ импорта:

– «Import via file»: Загрузить файл `ac-telemetry-dashboard.json`;

– «Import via URL»: Указать путь к файлу.

4) Указать параметры импорта:

– выбрать созданный источник данных Prometheus;

– задать имя дашборда;

– нажать кнопку «Import».

11.3.7. Проверить работоспособность

Убедиться, что панели дашборда отображают данные.

12. КОНФИГУРАЦИОННЫЕ ФАЙЛЫ СЦЕНАРИЕВ УСТАНОВКИ СРЕДЫ ФУНКЦИОНИРОВАНИЯ ППО И ИНФРАСТРУКТУРНЫХ КОМПОНЕНТОВ ППО

12.1. Конфигурационные файлы сценариев установки среды функционирования ППО и инфраструктурных компонентов ППО

12.1.1. Инвентарный файл inventories/hosts.yml

В инвентарном файле `inventories/hosts.yml` содержатся адреса серверов (имена хостов), на которые установлены (будут установлены) компоненты среды функционирования ППО и подсистемы ППО. Описание параметров инвентарного файла `inventories/hosts.yml` приведено в самом файле.

Файл сценария установки для установки среды функционирования ППО и инфраструктурных компонентов ППО на 1 сервере с доменным именем `ocs-app.local` имеет следующий вид:

```
---
all:
  children:
    ocs:
      vars:
        # Название кластера СУБД Postgres
        # patroni_cluster: ocs-primary

        # Виртуальные IP-адреса (virtual IP), которые будут присвоены
активному
        # серверу БД в основном кластере и StandBy Leader серверу БД в
резервном
        # кластере
        # keepalived_cluster_vip:
        #   ocs-primary: X.X.X.X
      children:
        # Сервера приложений ППО
        app:
          hosts:
            ocs-app.local:
        # Контент-сервера
        content:
```

```
          hosts:
        # СУБД Postgres
        postgresql:
          hosts:
            ocs-app.local:
        # Кеширующий DNS-сервер
        dnsmasq:
          children:
            app:
            content:
            redpanda:
        # Балансировщик сервисов "Nginx Web Server"
        nginx:
          children:
            app:
            content:
            pxe:
        # Система обнаружения сервисов "Consul" сервера приложений
        consul:
          children:
            # Сервера системы обнаружения сервисов "Consul"
            consul_servers:
              children:
                app:
            # Агенты системы обнаружения сервисов "Consul"
            consul_agents:
              children:
                # postgresql:
        # Система обнаружения сервисов "Consul" контент-сервера
        consul_content:
          children:
            content:
        # Средство управления конфигурациями сервисов "Consul
Template"
        consul_template:
          children:
            app:
            content:
        # Сервис гарантированной доставки сообщений "Redpanda"
        redpanda:
          children:
            app:
        valkey:
            # СУБД Valkey для хранения сессий
          children:
            app:
            # Valkey Sentinel обеспечивает высокую доступность СУБД
Valkey
        sentinel:
          children:
            app:
```

```
        # Приложение для синхронизации файлов Syncthing
        syncthing:
          children:
            app:
        # PXE сервер для загрузки по сети
        pxe:
          children:
            app:
          hosts:
```

12.1.2. Настройки сценариев установки среды функционирования ППО и инфраструктурных компонентов ППО в конфигурационных файлах config/vars/_vars.yml и config/subsystems/<название подсистемы>/vars/_vars.yml

В данных конфигурационных файлах задаются настройки следующих компонентов: Redpanda, Consul, СУБД Valkey и СУБД PostgreSQL. Конфигурационные файлы `_vars.yml` используются только в процессе установки.

Описание параметров конфигурационных файлов `_vars.yml` приведено в конфигурационных файлах в виде комментариев.

12.1.3. Настройки паролей и секретов компонентов среды функционирования ППО и инфраструктурных компонентов ППО в конфигурационных файлах config/secret.yml и config/subsystems/<название подсистемы>/secret.yml

В данных конфигурационных файлах задаются пароли и секреты следующих компонентов: Redpanda, Consul, СУБД Valkey и СУБД PostgreSQL.

13. КОНФИГУРАЦИОННЫЕ ФАЙЛЫ ППО (СЦЕНАРИЕВ УСТАНОВКИ ППО)

13.1. Общая информация о конфигурационных файлах ППО

ПРИМЕЧАНИЕ. Описание параметров конфигурационных файлов сценариев установки ППО и ППО приведено в конфигурационных файлах в виде комментариев.

Структура конфигурационных файлов ППО в общем виде приведена на рисунке (Рисунок 52). Жирным шрифтом выделены файлы, подлежащие редактированию. Редактирование параметров в остальных файлах не предполагается.

```
/var/ocs/config/
├── environments
│   └── <название окружения>
│       ├── vars
│       │   └── _vars.yml
│       └── config.yml
├── subsystems
│   └── <название подсистемы>
│       ├── applications
│       │   ├── <название сервиса>
│       │   └── vars
│       │       ├── services.yml
│       │       └── _vars.yml
│       ├── config.yml
│       ├── internal.yml
│       └── scopes.yml
├── config.yml
├── internal.yml
└── secret.yml
```

Рисунок 52

ППО содержит следующие типы конфигурационных файлов:

– конфигурационный файл ППО (`/vars/ocs/config/config.yml`);

– конфигурационные файлы подсистем ППО (`/vars/ocs/config/subsystems/<название подсистемы>/config.yml`);

– конфигурационные файлы с паролями и токенами компонентов среды функционирования ППО и инфраструктурных компонентов ППО (`/vars/ocs/config/secret.yml`);

– конфигурационные файлы сервисов (модулей) ППО (`/vars/ocs/config/subsystems/<название подсистемы>/applications/<название сервиса>/`).

В конфигурационном файле ППО содержатся настройки ППО.

В конфигурационных файлах подсистем содержатся настройки подсистем ППО. Также в конфигурационные файлы подсистем вынесены (могут быть вынесены) отдельные настройки сервисов ППО, которые может изменять администратор ППО. В данном случае в конфигурационном файле содержится секция с именем сервиса. Например, секция для сервиса `ocs-auth-accounts-users-api` выглядит следующим образом:

```
#-----------------------------------------------------------------------
---------
# Parameters for user accounts
#-----------------------------------------------------------------------
---------
ocs-auth-accounts-users-api:

  ##
  # The number of recently used passwords,
  # which system will store for forbidding use it for new password
creating.
  ##
  passwordHistoryDepth: 3

  ##
  # Maximum inactivity period 45 days.
  # If account not use system during this time, account will be
blocked.
  # Must be greater or equal to OIDC refresh token lifetime.
  ##
  maxAccountInactivityPeriod: "1080h"
```

ВНИМАНИЕ! Редактирование конфигурационных файлов сервисов не предполагается.

13.2. Общая информация о конфигурационных файлах сценариев установки ППО

Структура конфигурационных файлов сценариев установки ППО в общем виде приведена на рисунке (Рисунок 53). Жирным шрифтом выделены файлы, подлежащие редактированию. Редактирование параметров в остальных файлах не предполагается.

```
install-ac-mt
├── config
│   ├── environments
│   │   └── <название окружения>
│   ├── subsystems
│   │   └── <название подсистемы>
│   │       ├── applications
│   │       │   └── <название сервиса>
│   │       ├── vars
│   │       │   ├── services.yml
│   │       │   └── _vars.yml
│   │       ├── config.yml(или config.yml.j2)
│   │       ├── internal.yml
│   │       └── scopes.yml
│   ├── vars
│   │   └── _vars.yml
│   ├── config.yml.j2
│   ├── internal.yml
│   ├── yamllint.release.yml
│   └── secret.yml
└── inventories
    ├── <название окружения>
    └── hosts.yml
```

Рисунок 53

Сценарии установки ППО содержат следующие типы конфигурационных файлов:

– инвентарный файл `inventories/hosts.yml`;

– конфигурационный файл сценария установки ППО `config/vars/_vars.yml`;

– конфигурационные файлы сценариев установки подсистем ППО `config/subsystems/<название подсистемы>/vars/_vars.yml`;

– шаблон конфигурационного файла ППО (`config/config.yml.j2`);

– конфигурационные файлы подсистем ППО (config/subsystems/<название подсистемы>/config.yml);

– шаблоны конфигурационных файлов подсистем ППО (config/subsystems/<название подсистемы>/config.yml.j2);

– конфигурационные файлы сервисов (модулей) ППО (config/subsystems/<название подсистемы>/applications/<название сервиса>/);

– конфигурационный файл с паролями и токенами компонентов среды функционирования ППО и инфраструктурных компонентов ППО config/secret.yml.

13.2.1. Инвентарный файл inventories/hosts.yml

В инвентарном файле inventories/hosts.yml содержатся адреса серверов (имена хостов), на которые установлены (будут установлены) компоненты среды функционирования ППО, инфраструктурные компоненты ППО и подсистемы ППО.

Описание параметров инвентарного файла inventories/hosts.yml приведено в п. 12.1.1.

13.2.2. Общий конфигурационный файл сценариев установки config/vars/_vars.yml

Конфигурационный файл config/vars/_vars.yml является общим для всех подсистем и модулей ППО и содержит полный перечень общих параметров, относящихся к подсистемам и модулям ППО.

Конфигурационные файлы _vars.yml используются только в процессе установки, при этом конфигурационные файлы config.yml (config.yml.j2) используются как в процессе установки, так и в процессе эксплуатации ППО.

13.2.3. Конфигурационные файлы сценариев установки для подсистем ППО (файлы: config/subsystems/<название подсистемы>/vars/_vars.yml)

Конфигурационные файлы `_vars.yml` подсистем содержат параметры, относящиеся к конкретной подсистеме. Также данные файлы могут быть дополнены параметрами из общего конфигурационного файла, значения которых необходимо переопределить для заданной подсистемы.

Конфигурационные файлы `_vars.yml` в основном содержат настройки взаимодействия подсистем с компонентами среды функционирования ППО и инфраструктурными компонентами ППО. Располагаются в каталоге со сценариями установки по следующему пути:

```
config/subsystems/<название подсистемы>/vars/_vars.yml
```

Например, конфигурационный файл `vars.yml` для ПБ:

```
config/subsystems/auth/vars/_vars.yml
```

13.2.4. Шаблоны конфигурационных файлов ППО и подсистем ППО

На основе данных файлов в процессе установки ППО формируются конфигурационные файлы ППО и подсистем ППО. Значения параметров в шаблонах конфигурационных файлов подсистем ППО задаются администратором, а также сценариями установки на основе значений, заданных администратором в конфигурационных файлах `_vars.yml`.

Данные конфигурационные файлы располагаются по следующему пути:

```
config/config.yml.j2
config/subsystems/<название подсистемы>/config/services/config.yml.j2
```

Например, шаблон конфигурационного файла ПБ:

```
config/subsystems/auth/config/services/config.yml.j2
```

13.2.5. Конфигурационный файл с паролями и токенами компонентов среды функционирования ППО и инфраструктурных компонентов ППО config/secret.yml

В данном конфигурационном файле задаются пароли и токены Redpanda, Consul, СУБД Valkey, СУБД PostgreSQL, а также секретный ключ клиентов (сервисов) и ключ шифрования секретов, хранящихся в БД. При установке ППО данные конфигурационные файлы копируются на серверы приложений.

13.2.6. Конфигурационные файлы подсистем ППО

В данных конфигурационных файлах задаются значения параметров подсистем ППО. В отличие от шаблонов конфигурационных файлов подсистем ППО значения параметров задаются только администратором.

Данные конфигурационные файлы располагаются по следующему пути:

```
config/subsystems/<название подсистемы>/config/services/config.yml
```

Например, шаблон конфигурационного файла ПМ:

```
config/subsystems/appstore/config/services/config.yml.j2
```

13.2.7. Конфигурационные файлы сервисов ППО

Конфигурационные файлы сервисов располагаются в каталоге со сценариями установки по следующему пути:

```
config/subsystems/<название подсистемы>/applications/<название сервиса>/
```

Например, конфигурационные файлы сервиса `ocs-auth-adminconsole-ui` ПБ:

```
config/subsystems/auth/applications/ocs-auth-adminconsole-ui/
```

Описание параметров конфигурационных файлов сервисов приведено в самих конфигурационных файлах в виде комментариев.

ВНИМАНИЕ! Редактирование конфигурационных файлов сервисов ППО не предполагается.

13.2.8. Конфигурационные файлы окружений

В конфигурационных файлах окружения переопределяются параметры конфигурационных файлов, описанных в п. 13.2.1 – 13.2.7 для заданного окружения. Располагаются данные конфигурационные файлы в каталогах `config/environments/<название окружения>/` и `inventories/<название окружения>/`.

ВНИМАНИЕ! В конфигурационных файлах окружений не допускается использовать шаблоны конфигурационных файлов (файлов с расширением «.j2»).

Для переопределения параметра необходимо выполнить следующие действия:

– создать в каталоге `inventories/<название окружения>/` инвентарный файл `hosts.yml` по аналогии с файлом `inventories/hosts.yml` и задать в созданном файле требуемые значения параметров;

– создать в каталоге `config/environments/<название окружения>/` требуемый конфигурационный файл с учетом его расположения в каталоге `config`.

Например, для переопределения параметров конфигурационного файла `config/vars/_vars.yml` для окружения `release` должен быть создан следующий конфигурационный файл: `config/environments/release/config/vars/_vars.yml`;

– скопировать требуемый параметр (включая секцию, в которую входит параметр) из общего конфигурационного файла сценариев установки ППО или конфигурационного файла сценариев установки подсистем ППО;

– вставить скопированное значение в аналогичный конфигурационный файл для заданного окружения;

– задать требуемое значение параметра.

13.2.9. Порядок работы с конфигурационными файлами сценариев установки ППО

Параметры конфигурационных файлов сценариев установки применяются согласно приоритетам, заданным в таблице (Таблица 35).

Таблица 35

Типы конфигурационных файлов	Каталог (имя файла)	Порядок применения параметров (приоритет параметров)

Общие (для всех подсистем и модулей ППО) конфигурационные файлы (шаблоны конфигурационных файлов) сценариев установки ППО	`config/vars/_vars.yml`
	
	`config/config.yml.j2`
		1 (самый низкий приоритет)

Конфигурационные файлы сценариев установки подсистем ППО	`config/subsystems/<название подсистемы>/vars/` Например, `config/subsystems/auth/vars/`	2

Общие (для всех подсистем и модулей ППО) конфигурационные файлы сценариев установки ППО для заданного окружения	`config/environments/<название окружения>/vars/_vars.yml`
	
	`config/environments/<название окружения>/config.yml`
		3

Конфигурационные файлы сценариев установки подсистем ППО для заданного окружения	`config/environments/<окружение>/<название подсистемы>/vars/`	4 (самый высокий приоритет)

При установке ППО параметры конфигурационных файлов применяются в соответствии с порядком, приведенным в таблице (Таблица 35), т.е. сценарий установки обрабатывает сначала конфигурационные файлы в каталоге `config/vars/`, затем в каталоге `config/subsystems/<название подсистемы>/vars/` и т.д. Если, например, какой-либо параметр одновременно задан и в `config/vars/` и `config/subsystems/<название подсистемы>/vars/`,

ППО будет установлено со значением параметра, заданным в `config/subsystems/<название подсистемы>/vars/`.

Ниже описаны правила обработки сценариями установки ППО параметров, массивов и списков, если они одновременно заданы в нескольких конфигурационных файлах.

Правило обработки параметров: значение параметра в конфигурационном файле с более высоким приоритетом переопределяет значение параметра в конфигурационном файле с более низким приоритетом.

Пример параметра:

```
valkey_password: "example_valkey_password"
```

Правило обработки массивов: массив в конфигурационном файле с более высоким приоритетом переопределяет массив в конфигурационном файле с более низким приоритетом.

Пример массива:

```
Pg_hba_settings:
  - type: local  # Unix-socket access
    name: all
    database: all
    method: trust
  - type: host  # Localhost IPv4 access
    name: all
    database: all
    address: 127.0.0.1/32
    method: trust
  - type: host  # Localhost IPv6 access
    name: all
    database: all
    address: ::1/128
    method: trust
  - type: host  # Gitlab CI vbox-testing
    name: all
    database: all
    address: 172.17.0.0/16
    method: md5
```

Правило обработки списков: если список в конфигурационном файле с более низким приоритетом содержит новые элементы (которых не было в конфигурационном файле с более высоким приоритетом), они добавляются к исходному списку. Значение параметра в списке, содержащемся в конфигурационном файле с более высоким приоритетом, переопределяет значение параметра из списка, содержащегося в конфигурационном файле с более низким приоритетом.

Пример списка:

```
postgresql:
  dbname: example_db_name # database name
  port: 5432              # port
  user: example_user      # user
  password: ocs           # password
  extensions: ["pg_trgm", "pg_stat_statements", "pgcrypto", "pg_cron",
"auth_delay", "uuid-ossp"] # necessary extensions
```

14. ГАРАНТИЙНЫЕ ОБЯЗАТЕЛЬСТВА

Предприятие-изготовитель гарантирует работоспособность ППО в соответствии с заявленными характеристиками, предусмотренными настоящим документом, при соблюдении потребителем требований ЭД.

Предприятие-изготовитель проводит мониторинг общедоступных источников информации, публикующих сведения об уязвимостях, на предмет появления в них сведений об уязвимостях в компонентах ППО и принимает меры, направленные на устранение выявленных уязвимостей или исключающие возможность использования нарушителями выявленных уязвимостей.

Предприятие-изготовитель обеспечивает устранение уязвимостей посредством предоставления потребителям описания необходимых организационно-технических процедур, направленных на устранение выявленной уязвимости. Также предприятие-изготовитель, в рамках проведения работ по устранению выявленных уязвимостей, разрабатывает обновления ПО.

Предприятие-изготовитель не предоставляет гарантий или условий (явных или подразумеваемых законодательством Российской Федерации) относительно гарантий товарной пригодности, интегрируемости, годности к использованию для выполнения конкретных задач потребителя, отсутствия ошибок, возможности функционирования при использовании совместно с любым программным или аппаратным обеспечением.

В случае выявления в ППО ошибок и дефектов, свидетельствующих о несоответствии ППО ЭД и не являющихся уязвимостями ППО, предприятие-изготовитель по факту получения рекламации потребителя обязуется устранить ошибки и/или дефекты при выпуске обновленных версий ППО и уведомить об этом потребителей ППО.

Рекламации потребителя принимаются при условии, что дефект в ППО не вызван допущенными со стороны потребителя нарушениями в эксплуатации, хранении и транспортировке ППО.

Адрес предприятия-изготовителя для направления рекламаций: 420500, Республика Татарстан, Верхнеуслонский район, г. Иннополис, ул. Университетская, д. 7, офис 59, ОГРН 1161690087020.

ПЕРЕЧЕНЬ ТЕРМИНОВ И СОКРАЩЕНИЙ

Используемые в настоящем документе термины и сокращения приведены в таблице (Таблица 36).

Таблица 36

Термин/ Сокращение	Расшифровка
БД	База данных
ГИС	Государственная информационная система
ЗПС	Замкнутая программная среда
ИС	Информационные системы
НСД	Несанкционированный доступ
ОС	Операционная система
Партиционирование	Разделение таблицы на отдельные части (партиции) с целью улучшить производительность выполнения SQL запросов. Партиционирование позволяет разделить данные на более управляемые части, улучшая их доступность и обработку. Каждая партиция может быть независимо обработана, что упрощает и ускоряет выполнение запросов
ПБ	Подсистема безопасности
ПМ	Подсистема «Маркет»
ПО	Программное обеспечение
ПООС	Подсистема обновления ОС
ПСУ	Подсистема Сервис уведомлений
ПУ	Подсистема Платформа управления
ПУТ	Подсистема управления тенантами
ППО	Прикладное программное обеспечение «Аврора Центр»
Предприятие-изготовитель, предприятие-разработчик	Общество с ограниченной ответственностью «Открытая мобильная платформа» (ООО «Открытая мобильная платформа»)
Приложение	Приложением является: – мобильное приложение, функционирующее под управлением ОС Аврора/ОС Android; – приложение для ЭВМ, функционирующей под управлением ОС семейства Linux

Термин/ Сокращение	Расшифровка
РТК-Феникс	Доверенный репозиторий, обеспечивающий возможность применения безопасных библиотек свободного ПО в проектах разработки ПО. Разработан ООО «РТК ИТ»
ТУЗ	Техническая учетная запись
СЗИ	Средство защиты информации
СПО	Специальное программное обеспечение
СУБД	Система управления базами данных
СУДИР	Система управления доступом к информационным ресурсам
Токен	Аутентификационные данные, которые выдаются пользователю после успешной авторизации и являются ключом для доступа к службам
Типы портов	1. Внешний – доступ к данному типу портов осуществляется из-за пределов контролируемой зоны. Например, запросы от пользователей с ролью Пользователь Аврора Маркет. Доступ к данным портам имеет нарушитель; 2. Внутренний – доступ к данному типу портов может осуществляться только из контролируемой зоны. Данные порты используются для взаимодействия друг с другом сервисов ППО, компонентов среды функционирования ППО и инфраструктурных компонентов ППО, а таже для взаимодействия привилегированных пользователей с ППО
Устройство	Под устройством подразумевается мобильное устройство и/или ЭВМ, на которой функционируют соответствующие компоненты ППО
ЦОД	Центр обработки данных
ЭВМ	Электронно-вычислительная машина
ЭД	Эксплуатационная документация
API	Application Programming Interface – описание способов (набор классов, процедур, функций, структур или констант), которыми одна компьютерная программа может взаимодействовать с другой программой
CD-ROM	Compact Disc Read-Only Memory – разновидность компакт-дисков с записанными на них данными, доступными только для чтения
CDN	Подсистема доставки контента (Content Delivery Network)

Термин/ Сокращение	Расшифровка
Cookie	Небольшой фрагмент данных, отправленный веб-сервером и хранимый на ЭВМ пользователя. Веб-клиент (обычно веб-браузер) всякий раз при попытке открыть страницу соответствующего веб-сайта пересылает этот фрагмент данных веб-серверу в составе http-запроса
CSS3	Cascading Style Sheets 3 – спецификация CSS. Представляет собой формальный язык, реализованный с помощью языка разметки
DHCP	Dynamic Host Configuration Protocol – сетевой протокол, позволяющий сетевым устройствам автоматически получать IP-адрес и другие параметры, необходимые для работы в сети TCP/IP
DNS	Domain Name System - компьютерная распределенная система для получения информации о доменах
DVD	Digital Video Disc – оптический носитель информации, выполненный в форме диска, для хранения различной информации в цифровом виде
ECMAScript 5	Встраиваемый расширяемый не имеющий средств ввода-вывода язык программирования, используемый в качестве основы для построения других скриптовых языков
HTML5	HyperText Markup Language, version 5 – язык для структурирования и представления содержимого веб-страницы
HTTP	HyperText Transfer Protocol – протокол прикладного уровня передачи данных. Основой HTTP является технология «клиент-сервер», то есть предполагается существование потребителей (клиентов), которые инициируют соединение и посылают запрос, и поставщиков (серверов), которые ожидают соединения для получения запроса, производят необходимые действия и возвращают обратно сообщение с результатом
HTTPS	Hypertext Transfer Protocol Secure – расширение протокола HTTP для поддержки шифрования в целях повышения безопасности. Данные в протоколе HTTPS передаются поверх криптографических протоколов SSL или TLS

Термин/ Сокращение	Расшифровка
IP	Internet Protocol - основной протокол сетевого уровня, использующийся в Интернете и обеспечивающий единую схему логической адресации устройств в сети и маршрутизацию данных
ISO-образ	Образ оптического диска, содержащий файловую систему стандарта ISO 9660
JSON	JavaScript Object Notation – текстовый формат обмена данными, основанный на JavaScript
MTP	Media Transfer Protocol – аппаратно-независимый протокол, основанный на PTP
NFS	Network File System – протокол сетевого доступа к файловым системам, позволяющий монтировать (подключать) удаленные файловые системы через сеть. За основу взят протокол вызова удалённых процедур (ONC RPC)
Nginx	Веб-сервер и почтовый прокси-сервер, работающий на Unix-подобных ОС
OIDC	OpenID Connect – уровень аутентификации OAuth 2.0, инфраструктуры авторизации. Контролируется OpenID Foundation
PXE	Preboot eXecution Environment – среда для загрузки ЭВМ с помощью сетевой карты без использования локальных носителей (жестких дисков, компакт-дисков и других устройств)
RPM-пакет	Файл формата `.rpm`, позволяющий устанавливать, удалять и обновлять приложения на устройствах
SIEM-система	Система для мониторинга, анализа и управления событиями безопасности
SMTP	Simple Mail Transfer Protocol – сетевой протокол, предназначенный для передачи электронной почты в сетях TCP/IP
SSH	Secure SHell – сетевой протокол прикладного уровня, позволяющий производить удаленное управление ОС и туннелирование TCP-соединений (например, для передачи файлов)
TCP	Transmission Control Protocol – протокол транспортного уровня, гарантирующий целостность передаваемых данных и уведомление отправителя о результатах передачи

Термин/ Сокращение	Расшифровка
TLS	Transport Layer Security – криптографический протокол, обеспечивающий защищенную передачу данных между узлами в сети Интернет
URL	Uniform Resource Locator – единообразный локатор (определитель местонахождения) ресурса

ЛИСТ РЕГИСТРАЦИИ ИЗМЕНЕНИЙ

№ версии документа	Содержание изменения	ФИО	Дата
1.0	Начальная версия	Фомин П. Гребенюк Б.	10.09.2026 г.
1.1	Внесены изменения	Пенджиев М. Гребенюк Б.	17.09.2026 г.
1.2	Внесены изменения	Пенджиев М. Гребенюк Б.	12.10.2026 г.
1.3	Внесены изменения	Пенджиев М. Гребенюк Б.	16.10.2026 г.
1.4	Внесены изменения	Пенджиев М. Гребенюк Б.	05.11.2026 г.