Руководство администратора

Установка

04.12.2024

ВВЕДЕНИЕ 3
ТЕРМИНЫ И ОПРЕДЕЛЕНИЯ 4
ОБЩИЕ СВЕДЕНИЯ 5
Назначение ПО 5
Основные функции 5
Основные компоненты 5
Архитектура 6
СИСТЕМНЫЕ ТРЕБОВАНИЯ 8
Требования к платформе 8
Требования к DNS 10
Требования к корпоративному каталогу LDAP 11
Требования к серверу SMTP 11
УСТАНОВКА YUCHAT 12
Предварительная настройка 12
ОС Ubuntu/Debian 12
ОС Astra Linux Орел 14
Установка комплекса 15
НАСТРОЙКА ПЕРЕМЕННЫХ ИНФРАСТРУКТУРЫ 17
Настройка BLOB-хранилища 17
Настройка БД 17
Настройка RabbitMQ 17
Настройка Redis 17
НАСТРОЙКА СЕРВЕРА 18
Подключение SMTP-сервера 18
API и Secret ключи 18
Настройка APNS 18
Настройка FCM 20
Эксплуатация сервера 21
ОБНОВЛЕНИЕ СЕРВЕРА 22
Ручное обновление 22
ИНСТРУКЦИЯ ПО РАЗВЕРТЫВАНИЮ В KUBERNETES 24
Требования к окружению 24
Требования к инструментарию 24
Установка 26
Подготовка HelmCharts 28

ВВЕДЕНИЕ

Руководство предназначено для администраторов ПО «Платформа корпоративных коммуникаций YuChat» (далее – ПКК «YuChat», YuChat, система). В нем содержатся сведения, необходимые для установки и настройки системы.

Служба технической поддержки. Связаться со службой технической поддержки можно по электронной почте support@yuchat.ai.

Сайт в интернете. Информацию о продукте компании «Unison Technologies» представлена на сайте https://yuchat.ru.

ТЕРМИНЫ И ОПРЕДЕЛЕНИЯ

AD	Active Directory — служба каталогов корпорации Microsoft для операционных систем семейства Windows Server
API	Application Programming Interface — интерфейс для взаимодействия программ и приложений
APNS	Apple Push Notification Service — сервис push-уведомлений Apple
FCM	Firebase Cloud Messaging — служба, которая упрощает обмен сообщениями между мобильными приложениями и серверных приложений
JSON	JavaScript Object Notation — текстовый формат обмена данными
SMTP	Сетевой протокол, предназначенный для передачи электронной почты в сетях TCP/IP
STUN	Сетевой протокол для определения внешнего IP-адреса, используемый для установления соединения UDP между двумя хостами в случае, если они оба находятся за маршрутизатором NAT
TLS	Протокол защиты транспортного уровня
ВКС	Видео- и конференцсвязь
Кэш	Промежуточный буфер с быстрым доступом, содержащий часто используемую информацию

ОБЩИЕ СВЕДЕНИЯ

Назначение ПО

ПКК YuChat разработан для обеспечения стабильной и высококачественной связи среди сотрудников компании, а также для уменьшения вероятности утечек информации путем переноса коммуникационных каналов с интернета в границы внутренних корпоративных сетей.

Основные функции

YuChat реализует следующие функции:

- быстрый обмен пользователей текстовыми сообщениями и файлами с помощью мобильных устройств и веб-клиента на ПК в рамках персональных и групповых чатов;
- осуществление персональных и групповых аудио- и видеозвонков;
- запись, расшифровка и хранение проведенных аудио- и видеозвонков;
- индикация присутствия и текущей активности пользователя в системе

Основные компоненты

ПКК YuChat включает следующие отдельно устанавливаемые компоненты:

- основное серверное приложение (управляет данными в системе)
- приложение для записи видео- и аудио-заметок;
- приложение для пост-обработки записей аудио- и видеозвонков;
- комплекс из трех приложений для обеспечения голосовой и видеосвязи между пользователями (WebRTC);
- приложение для доставки уведомлений и текстовых сообщений в реальном времени;
- мобильные приложения (iOS, Android);
- десктоп-приложение;
- веб-приложение;

Управление комплексом осуществляется с помощью веб-интерфейса — консоли администратора, которая предоставляет возможности для настройки YuChat и контроля функционирования приложения.

Архитектура

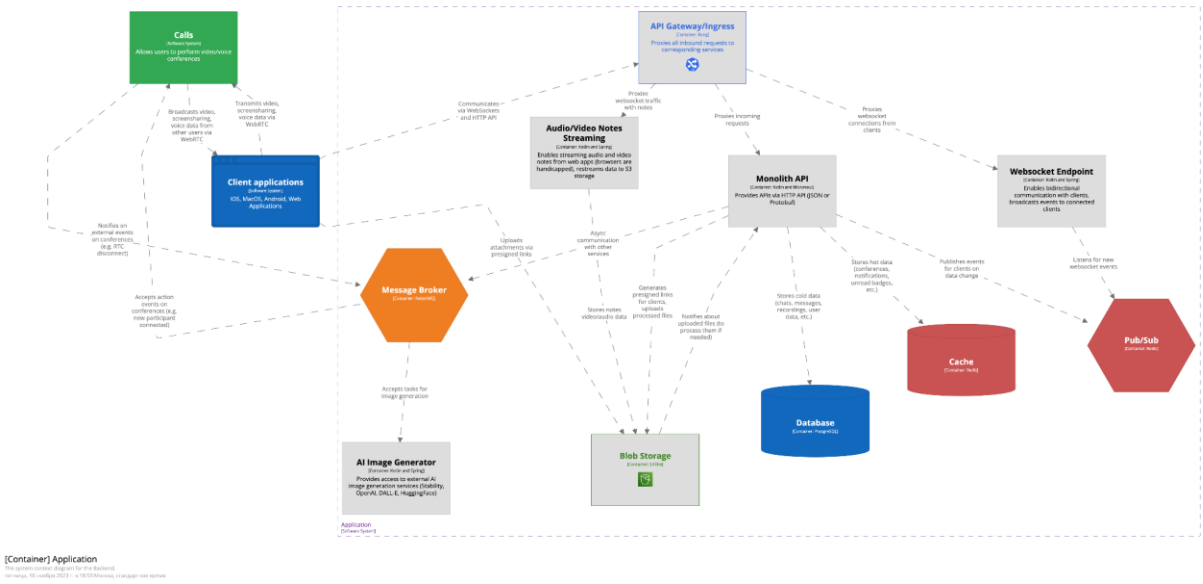
Важно! В данном документе рассматривается неотказоустойчивая конфигурация изделия. Для получения сведений о вариантах отказоустойчивой конфигурации обратитесь к разработчику.

Серверная часть YuChat основана на микросервисной архитектуре с использованием контейнеризации на основе Docker/Docker Swarm.

Для всех вариантов развертывания системы серверная часть размещается в сети Интернет и содержит в себе следующие контейнеры:

- ws-endpoint-service (сервис для подключения и обмена информацией с клиентскими приложениями в реальном времени с помощью websocket)
- web (веб-приложение)
- nginx-static (вспомогательные веб-страницы)
- mediahub (управление состоянием аудио и видеозвонков)
- media-composer (пост-обработка записей аудио и видеозвонков)
- freeswitch (сервис микширования аудио треков)
- webrtc (сервис обмена видео трафиком)
- backend (основной сервис управления данными в системе)
- avnotes-streaming (сервис приема и конвертации аудио и видео заметок)
- postgresql (БД для хранения данных системы)
- traefik (точка входа в кластер, reverse proxy)
- rabbitmq (брокер обмена сообщениями между сервисами)
- rabbitmq_exporter (сервис для экспорта статистики из rabbitmq в prometheus)
- postgresql_exporter (сервис для экспорта статистики из postgresql в prometheus)
- prometheus (сервис хранения статистики сервисов)
- redis (сервис кэширования часто используемых данных)
- redis_prometheus (сервис для экспорта статистики из redis в prometheus)
- minio (хранилище файлов)

Типовая схема развертывания YuChat размещена ниже (рис. 1)



[Container] Application

Диаграмма развертывания (рис. 1)

СИСТЕМНЫЕ ТРЕБОВАНИЯ

Требования к платформе

Минимальные системные требования к аппаратным платформам в зависимости от количества пользователей:

Медиа-сервер (звонки):

Количество участников	**CPU (vCPU)**	**RAM (GB)**	**Пропускная способность (MB/s)**
10	2	4	2.7 MB/s
100	12	16	31.6 MB/s
1000	48	72	316 MB/s
5000	128	190	1580 MB/s

Основной сервер (API):

Количество участников	**CPU (vCPU)**	**RAM (GB)**
10	8	16
100	12	32
1000	48	72
5000	64	128

Примечание. Объем SSD взят из расчета глубины хранения пользовательских данных (3 Гб) и записей звонков (7 Гб) за год. Данные по требуемому месту могут значительно отличаться от расчетных при более активном использовании комплекса или при условии использования подробного журналирования.

Минимальные системные требования к основному серверу для установки подсистем (без отказоустойчивости):

Элемент	Параметры
Процессор	4 ядра, частота не менее 2.60 ГГц
Оперативная память	16 ГБ
Операционная система	Ubuntu 22.04 LTS/20.04 LTS; Astra Linux Special Edition 1.6 и 1.7
Жесткий диск	Не менее 200 ГБ
Предустановленное ПО	Docker-ce версии 20.10.23; PostgreSQL версии 12 и выше; Redis версии 7.1 и выше; RabbitMQ версии 3.10.0 и выше; MinIO или другое S3-совместимое хранилище;

Минимальные системные требования к серверу для установки подсистем звонков (без отказоустойчивости):

Элемент	Параметры
Процессор	8 ядер, частота не менее 2.60 ГГц
Оперативная память	16 ГБ
Операционная система	Ubuntu 22.04 LTS/20.04 LTS; Astra Linux Special Edition 1.6 и 1.7
Жесткий диск	Не менее 500 ГБ
Предустановленное ПО	Docker-ce версии 20.10.23;

Распознавание речи (записи видео- и аудиозвонков): Для распознавания речи без отправки в сторонние сервисы можно использовать видеокарту или сервис Яндекс. От мощности видеокарты зависит скорость распознавания и постобработки записей. Минимальная поддерживаемая: Nvidia GeForce 3080. В этом случае данные не будут переданы в сервисы Яндекса для распознавания речи и получения заголовка записи.

Требование к операционной системе: Серверы поддерживают любую ОС семейства Linux, на который устанавливается Docker 20.10.23.

Примечание. Серверы поддерживают ОС Astra Linux 2.12.43 Common Edition «Орёл». Требование к ПО контейнеризации: Docker: 20.10.23 (настоятельно рекомендуется установка из репозитория docker).

Требование к синхронизации времени: Необходим установленный и настроенный локальный сервер NTP с уровнем stratum не ниже 15.

Для использования веб-интерфейса нужны, как минимум, следующие версии браузеров:

Браузер	Версия
Chrome	118
Chromium	118
Yandex Browser	23.9
Firefox	113
Safari	16.6.1

Для использования десктоп-интерфейса рекомендуется использовать персональные компьютеры с операционными системами, перечисленными в таблице ниже:

ОС	Версия
MacOS	12
Windows	10

Требования к DNS

Для корректной работы YuChat требуется DNS-имя для основного сервера, разрешаемое в сети Интернет и ссылающиеся на внешний IP-адрес публикации сервера. Рекомендуется имя третьего уровня, например yuchat.mydomain.tld.

Во внутренней сети компании DNS-имя должно разрешаться во внутренний IP-адрес сервера. Допускается настройка средствами ОС linux (служба systemd-resolved) с преобразованием во внутренней сети компании имен во внутренний IP-адрес. Требования к DNS-имени STUN/TURN сервера аналогичны требованиям к DNS имени сервера.

Требования к корпоративному каталогу LDAP

При интеграции YuChat с корпоративным каталогом на базе Microsoft Active Directory требуется создание учетной записи с правами «Domain Users» и контейнера «deleted objects» . Стандартной практикой предоставления доступа пользователей к YuChat является создание группы пользователей YuChat в Active Directory. Тип группы — «Security», видимость группы — «Universal». При интеграции YuChat с корпоративным каталогом на базе LDAP-совместимого сервера требуется создание учетной записи с правами чтения каталога. При использовании каталога AD LDS авторизация пользователей осуществляется только по коду на email.

Требования к серверу SMTP

Для возможности отправки кодов аутентификации устройства пользователя требуется создание на почтовом сервер учетной записи, под которой будет производиться отправка электронной почты.

УСТАНОВКА YUCHAT

Предварительная настройка

Для корректной работы сервера выполните предварительную настройку.

Внимание! Установку YuChat должен осуществлять пользователь Linux с опытом администрирования. Предварительная настройка зависит от ОС.

ОС Ubuntu/Debian

Для предварительной настройки при использовании ОС Ubuntu/Debian:

1. Установите ОС Ubuntu 22.04 LTS или Ubuntu 20.04 LTS. Воспользуйтесь официальным источником для установки дистрибутива: https://ubuntu.com/download/server

Внимание! Во время установки ОС выделите под корневой «/» раздел 32 Гб, под SWAP раздел выделить 8 Гб, оставшееся место выделите под раздел «/var/lib/docker».

2. Удалите пакеты snapd и ufw с помощью команды: apt autoremove --purge snapd ufw
3. Установите программное обеспечение Docker. Для установки воспользуйтесь официальным источником: https://docs.docker.com/install/linux/docker-ce/ubuntu/

Внимание! Если ПО Docker распаковано из пакета snapd, удалите его и выполните установку из официального источника. Пример кода для установки Docker:

```
apt-get remove docker docker-engine docker.io containerd runc
apt-get update
apt-get install ca-certificates curl gnupg lsb-release
curl -fsSL https://download.docker.com/linux/ubuntu/gpg | gpg --
dearmor -o /usr/share/keyrings/docker-archive-keyring.gpg
echo \ "deb [arch=$(dpkg --print-architecture)
signedby=/usr/share/keyrings/docker-archive-keyring.gpg]
https://download.docker.com/linux/ubuntu \ $(lsb_release -cs)
stable" | tee /etc/apt/sources.list.d/docker.list > /dev/null
apt-get update
apt-get install docker-ce docker-ce-cli containerd.io
```

4. Установите дополнительное ПО (см. ниже).

Для установки дополнительного ПО:

1. Выполните установку NTP-сервера с помощью команды:

```
apt install chrony
```

Если имеются источники точного времени внутри компании, в файл /etc/chrony/chrony.conf внесите серверы NTP в виде:

```
server ntp1.local
server ntp2.local
server ntp3.local
```

Пример кода:

```
systemctl enable chrony
systemctl restart chrony
```

Для проверки подключения к NTP-серверам используйте следующую команду:

```
chronyc sources -v
```

2. Укажите параметры хранения журналов в Docker в каталоге /etc/docker/daemon.json:

```
{
  "log-driver": "json-file",
  "log-opts": {
    "max-size": "1g"
  }
}
```

Выполните:

```
systemctl restart docker
```

ОС Astra Linux Орел

Для предварительной настройки при использовании ОС Astra Linux Орел:

1. Установите ОС Astra Linux Орел. Во время установки на шаге выбора «Выбор программного обеспечения» выделите Базовые средства, Средства удаленного доступа SSH.
2. Установите Docker помощью команды:

```
apt install docker.io
```

3. Установите дополнительное ПО (см. ниже).

   Выполните установку NTP-сервера с помощью команды:

```
apt install chrony
```

Если имеются источники точного времени внутри компании, в файл /etc/chrony/chrony.conf внесите серверы NTP.

Удалите или закомментируйте строку pool и укажите свои сервера.

Пример:

```
server ntp1.local
server ntp2.local
server ntp3.local
```

Перезапустите службу для применения изменений:

```
systemctl restart chrony
```

Установка комплекса

Все описанные действия должны выполняться на узле-менеджере кластера в случае если узлов несколько и на самом узле если он один.

1. Разархивируйте комплект поставки (yuchat.zip)

```
unzip yuchat.zip
```

2. Используйте переданные разработчиком учетные данные для входа в регистр образов.

```
docker login -u Login -p Password registry.yuchat.ai
```

3. Отредактируйте конфигурационные файлы сервисов:

    - В папке configuration находятся конфигурационные файлы под каждый из сервисов. В них можно настроить конфигурацию под свое окружение, но желательно придерживаться дефолтной конфигурации.
    - В файлах в папке env заполните все переменные окружения соответствующими значениями из предустановленного ПО (PostgreSQL, Redis, RabbitMQ).

4. Добавьте роли на узлы
   Роль main указывается на узле, куда будут установлены основные сервисы API:

```
docker node update --label-add yuchat.role.main=true
```

   Роль media указывается на узле, куда будут установлены сервисы для обеспечения видео- и аудиозвонков:

```
docker node update --label-add yuchat.role.media=true
```

   Роль storage указывается на узле, куда будут установлены postgresql и minio:

```
docker node update --label-add yuchat.role.storage=true
```

   Роль exchange указывается на узле, куда будут установлены redis и rabbitmq:

```
docker node update --label-add yuchat.role.exchange=true
```

5. Переведите один из узлов (или единственный) в режим swarm:

```
docker swarm init
```

6. Запустите размещение комплекса на узлах (узле):

```
docker stack deploy -c stack.yml yuchat
```

7. Проверьте состояние установки (она может занимать до 5 минут):

```
docker stack ps
```

   В случае, когда один или несколько контейнеров имеют статус Error, обратитесь к разработчику для диагностики.

НАСТРОЙКА ПЕРЕМЕННЫХ ИНФРАСТРУКТУРЫ

Настройка BLOB-хранилища

S3-совместимое хранилище используется для хранения прикрепленных к сообщениям файлов, аудио и видеозаметок, записей звонков.

В файле **shared.env** необходимо указать следующие переменные окружения:

```
MINIO_ROOT_USER // ключ доступа и username для minio
MINIO_ROOT_PASSWORD // секретный ключ и password для minio
MINIO_HOST // адрес сервиса
MINIO_PORT // порт сервиса
```

Настройка БД

PostgreSQL используется для хранения большинства данных в системе.

В файле **shared.env** необходимо указать следующие переменные окружения:

```
DATASOURCE_URL // адрес БД
POSTGRES_USER // имя пользователя
POSTGRES_PASSWORD // пароль пользователя
DATASOURCE_SCHEMA  // схема БД
```

Настройка RabbitMQ

RabbitMQ используется в качестве брокера сообщений.

В файле **shared.env** необходимо указать следующие переменные окружения:

```
RABBITMQ_URI // полный адрес с протоколом amqp
RABBITMQ_USERNAME // имя пользователя
RABBITMQ_PASSWORD // пароль пользователя
```

Настройка Redis

Redis используется для кеширования данных в системе.

В файле **shared.env** необходимо указать следующие переменные окружения:

```
REDIS_HOST // адрес БД
REDIS_PASSWORD // пароль пользователя
REDIS_PORT  // порт сервиса
```

НАСТРОЙКА СЕРВЕРА

Подключение SMTP-сервера

SMTP-сервер необходим для рассылки уведомлений и писем с ссылками для входа по электронной почте.

В файле **backend.env** необходимо указать следующие переменные окружения:

```
SMTP_HOST // адрес сервера
SMTP_PORT // порт сервера. Обычно 25, 587 или 465. Номер порта зависит от типа защищенного соединения
SMTP_USERNAME // имя пользователя
SMTP_PASSWORD // пароль пользователя
SMTP_TLS_ENABLED // использовать ли TLS при подключении
```

API и Secret ключи

SMTP-сервер необходим для рассылки уведомлений и писем с ссылками для входа по электронной почте.

В файле **backend.env** необходимо указать следующие переменные окружения:

```
YACLOUD_API_KEY // используется для Yandex AI - Speechkit и YandexGPT
JWT_SHARED_SECRET // используется для кодирования jwt токенов пользователей, random string
```

Настройка APNS

Сервис Apple Push Notification System используется для рассылки оповещений пользователям YuChat на устройства с IOS.

Для корректной работы необходимо получить ключ в Apple.

Регистрация в Apple Developer Program

Чтобы получить доступ к APNs, вам необходимо быть членом Apple Developer Program. Если вы ещё не зарегистрированы, посетите официальный сайт Apple Developer и следуйте инструкциям для регистрации.

Доступ к Apple Developer Center

После регистрации или входа в свой аккаунт Apple Developer перейдите на Apple Developer Center.

Переход в Certificates, Identifiers & Profiles

В Apple Developer Center выберите раздел "Certificates, Identifiers & Profiles". Это централизованный интерфейс для управления сертификатами, идентификаторами приложений, профилями разработки и другими важными аспектами разработки под iOS и macOS.

Создание Certificate Signing Request (CSR)

На вашем Mac откройте приложение "Keychain Access" (Связка ключей).

Выберите в меню "Keychain Access" > "Certificate Assistant" > "Request a Certificate from a Certificate Authority".

Введите ваш адрес электронной почты и имя для сертификата, затем выберите "Saved to disk" и следуйте инструкциям для создания CSR.

Создание нового сертификата APNs в Apple Developer Center

Вернитесь в раздел "Certificates, Identifiers & Profiles" и выберите "Identifiers".

Найдите и выберите ваше приложение, для которого вы хотите настроить APNs.

В разделе Push Notifications следуйте инструкциям для создания нового сертификата APNs, используя созданный ранее CSR файл.

Загрузка и завершение создания сертификата

Загрузите CSR файл в Apple Developer Center. После обработки вашего запроса, Apple предоставит возможность скачать новый сертификат APNs.

Скачайте и сохраните сертификат

Далее укажите полученные данные в конфигурационном файле **backend-config.yml:**

```
apns:
  MACOS:
    app-id: "" #
    voip-app-id: ""
    key-id: ""
    team-id: ""
    signing-key-file-path: "AuthKey.p8"
    host-type: PRODUCTION
  IOS:
    app-id: ""
    voip-app-id: ""
    key-id: ""
    team-id: ""
    signing-key-file-path: "AuthKey.p8"
    host-type: PRODUCTION
```

Настройка FCM

Сервис FCM используется для рассылки оповещений пользователям YuChat на устройства с Android. Для настройки воспользуйтесь следующей инструкцией:

Перейти на сайт Firebase Console

Откройте браузер и перейдите на Firebase Console. Это официальная платформа для управления всеми сервисами Firebase, включая Cloud Messaging.

Вход в аккаунт Google

Если вы еще не вошли в свой аккаунт Google, вам будет предложено это сделать. Firebase требует Google-аккаунт для аутентификации.

Создание нового проекта

Нажмите на кнопку "Добавить проект" или "Create project" (в зависимости от языка интерфейса).

Введите название проекта. Это имя будет использоваться для идентификации вашего проекта в Firebase.

Примите условия Firebase, если это требуется.

Настройка проекта

Выберите страну/регион вашей организации. Это важно для соответствия местным законам о данных.

Определите, хотите ли вы включить или отключить Google Analytics для вашего проекта. Google Analytics предоставляет дополнительные аналитические возможности, но это необязательно для использования FCM.

Создание проекта

После заполнения всех необходимых данных нажмите на кнопку "Создать проект" или "Create project". Этот процесс может занять несколько минут.

После завершения создания проекта вы будете перенаправлены на страницу обзора проекта.

Получение Web API Key

В обзоре проекта найдите и скопируйте ваш Web API Key.

Этот ключ необходим для идентификации вашего сервера при обращении к сервисам Firebase, включая FCM.

Сохраните ключ в рабочей директории под именем fcm_key.json.

Эксплуатация сервера

Сервисы поддерживают экспорт журналов и статистических данных во внешние системы хранения и просмотра (Prometheus, Grafana / ElasticSearch, Kibana). Пакет поставки по умолчанию включает в себя Prometheus/Grafana. При наличии собственной инфраструктуры и необходимостью интеграции обратитесь к разработчику за консультациями.

ОБНОВЛЕНИЕ СЕРВЕРА

Ручное обновление

Подготовка

Проверьте состояние кластера

Убедитесь, что ваш кластер Docker Swarm работает корректно. Это можно сделать, используя команду:

```
docker node ls
```

Эта команда покажет все узлы в вашем кластере и их состояние.

Выясните, какие сервисы нужно обновить

Используйте команду:

```
docker service ls
```

Это позволит вам увидеть все запущенные сервисы и их текущие образы.

2. Обновление Образов

Получение новых версий образов

Обновите образы, которые вы хотите использовать. Это может быть сделано через pull новых версий с Docker Hub или вашего частного реестра. Например:

```
docker pull myimage:latest
```

Замените myimage:latest на название и тег обновленного образа.

Обновление сервиса в Docker Swarm

Используйте команду docker service update. Например, чтобы обновить сервис с новой версией образа, используйте:

```
docker service update --image backend:latest backend
```

Замените myservice на имя сервиса, который вы обновляете, и myimage:latest на новую версию образа.

При необходимости, вы можете добавить дополнительные параметры к команде для конфигурации процесса обновления, такие как --update-delay для задания задержки между обновлениями отдельных реплик.

3. Мониторинг и Верификация

Проверьте статус обновления

Используйте команду:

```
docker service ps backend
```

Это покажет статус задач сервиса и поможет убедиться, что обновление прошло успешно.

Логирование и отладка

Если возникают проблемы в процессе обновления, проверьте логи сервиса:

```
docker service logs backend
```

Это поможет определить причины возникновения ошибок или неполадок.

4. Пост-обновление

Тестирование функциональности

После обновления важно провести тестирование, чтобы убедиться, что все компоненты работают как ожидается.

Проверьте доступность и работоспособность обновленного сервиса.

ИНСТРУКЦИЯ ПО РАЗВЕРТЫВАНИЮ В KUBERNETES

Требования к окружению

- PostgreSQL 14+
- RabbitMQ 3.11+
- Redis 7+
- Meilisearch 1.8+
- Weaviate 1.25+
- MinIO 2023-10+
- SMTP-сервер

Опционально:

- Установленная на GPU-кластере LLM с API совместимым с OpenAI / ключ OpenAI / ключ Yandex Cloud Foundation Models (для суммаризации и других AI-функций)
- Установленный на GPU-кластере Whisper / ключ Yandex Foundation Models (для speech-to-text)

Все компоненты окружения могут быть размещены как в кластере так и вне его.

Требования к инструментарию

- ArgoCD 2.10+

Подготовка

PostgreSQL:

Создайте базу данных `yuchat_db`.

Создайте схему `yuchat`.

Создайте пользователя (ТУЗ) с правами `read/write` на эту базу данных и схему.

CREATE DATABASE yuchat_db;

RabbitMQ:

Создайте пользователя (ТУЗ) с правами `configure/read/write` на виртуальный хост `/`.

MinIO:

Создайте пользователя (ТУЗ) с правами на создание бакетов.

Настройте amqp нотификации с параметрами:
MINIO_NOTIFY_AMQP_ENABLE_PRIMARY: "on"
MINIO_NOTIFY_AMQP_URL_PRIMARY: "amqp://username:password@rabbitmq_host:5672"
MINIO_NOTIFY_AMQP_EXCHANGE_PRIMARY: "yuchat.aws.file.events.ex"
MINIO_NOTIFY_AMQP_EXCHANGE_TYPE_PRIMARY: "fanout"
MINIO_NOTIFY_AMQP_ROUTING_KEY_PRIMARY: "#"
MINIO_NOTIFY_AMQP_DURABLE_PRIMARY: "true"
MINIO_NOTIFY_AMQP_AUTO_DELETED_PRIMARY: "false"

На Minio должен быть настроен https!

Redis:

Создайте пользователя (ТУЗ).
Для Redis можно использовать стандартного пользователя, поскольку Redis обычно не использует отдельных учетных записей пользователей.

SMTP сервер:

Установите и настройте SMTP сервер для отправки почты (например, для подтверждения регистрации пользователей).
Убедитесь, что SMTP сервер доступен и настроен с параметрами `SMTP_HOST`, `SMTP_PORT`, `SMTP_USERNAME`, `SMTP_PASSWORD`.

Публичный IP адрес:

Подготовьте публичный IP адрес с портом 3478, на который будет создан gateway для обеспечения связи с STUN/TURN серверами.

Создание домена для Yuchat:

Зарегистрируйте домен, на котором будет развернут Yuchat.
Например: `yuchat.yourdomain.com`.

Установка

1. Создайте namespace yuchat:

```
kubectl create namespace yuchat
```

2. Создайте секрет с TLS сертификатом для домена и субдоменов:

Генерация и установка секрета:

```
kubectl create secret tls yuchat-tls --cert=/path/to/tls.crt --
key=/path/to/tls.key -n yuchat
```

3. Создайте секрет с типом docker-registry для доступа к репозиторию с образами:

```
kubectl create secret docker-registry yuchat-regcred \\
      --docker-server=https://registry.unison.team \\
      --docker-username= <обратитесь к поставщику за учетными данными> \\
      --docker-password= <обратитесь к поставщику за учетными данными> \\
      -n yuchat
```

4. Создайте StorageClass:

Пример YAML для StorageClass:

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: yuchat-storage
provisioner: kubernetes.io/aws-ebs
parameters:
  type: gp2
```

5. Настройка доступа в ArgoCD до репозитория с чартами:

В ArgoCD добавьте репозиторий:

```
    argocd repo add <https://registry.unison.team/yuchatcharts> \\
      --username <обратитесь к поставщику за учетными данными> \\
      --password <обратитесь к поставщику за учетными данными>
```

6. Создайте секреты для RabbitMQ, Redis, MinIO, PostgreSQL, SMTP и Stunner:

Redis:

```
kubectl create secret generic redis-creds-secret \\
  --from-literal=REDIS_HOST=redis.yuchat.yourdomain.com \\
  --from-literal=REDIS_PASSWORD=your_password \\
  --from-literal=REDIS_PORT=6379 \\
  -n yuchat
```

RabbitMQ:

```
kubectl create secret generic rabbitmq-creds-secret \\
  --from-literal=RABBITMQ_URI=amqp://rabbitmq.yuchat.yourdomain.com:5672 \\
  --from-literal=RABBITMQ_USERNAME=yuchat_user \\
  --from-literal=RABBITMQ_PASSWORD=your_password \\
  --from-literal=RABBITMQ_HOST=rabbitmq.yuchat.yourdomain.com \\
  --from-literal=RABBITMQ_VHOST=\ \\
  -n yuchat
```

MinIO:

```
kubectl create secret generic minio-creds-secret \\
  --from-literal=CLOUD_ACCESS_KEY=your_access_key \\
  --from-literal=CLOUD_SECRET_KEY=your_secret_key \\
  --from-literal=S3_URL=https://minio.yuchat.yourdomain.com \\
  --from-literal=S3_DOMAIN=minio.yuchat.yourdomain.com \\
  -n yuchat
```

PostgreSQL:

```
kubectl create secret generic postgresql-creds-secret \\
  --from-literal=DATASOURCE_USERNAME=yuchat_user \\
  --from-
literal=DATASOURCE_URL=jdbc:postgresql://postgres.yuchat.yourdomain.com:543
2/yuchat_db \\
  --from-literal=DATASOURCE_PASSWORD=your_password \\
  --from-literal=DATASOURCE_SCHEMA=yuchat \\
  -n yuchat
```

SMTP сервер:
Если отсутствуют username и password от сервера, установите SMTP_USERNAME и SMTP_PASSWORD в "".

```
kubectl create secret generic smtp-creds-secret \\
  --from-literal=MAIL_SMTP_HOST=smtp.yourdomain.com \\
  --from-literal=MAIL_SMTP_PORT=25 \\
  --from-literal=MAIL_SMTP_USERNAME=smtp_user \\
  --from-literal=MAIL_SMTP_PASSWORD=smtp_password \\
  -n yuchat
```

Stunner:

```
kubectl create secret generic stunner-creds-secret \\
  --from-literal=STUNNER_USERNAME=stunner_user \\
  --from-literal=STUNNER_PASSWORD=stunner_password \\
  --from-literal=STUNNER_PORT=3478 \\
  --from-literal=STUNNER_HOST=<public_ip> \\
  -n yuchat
```

7. Создайте секрет с переменными для backend сервисов:

```
kubectl create secret generic backend-secret \\
  --from-literal=JWT_SHARED_SECRET=<random_string_256_bits> \\
  --from-literal=MEILISEARCH_MASTER_KEY=<random_string> \\
  -n yuchat
```

Подготовка HelmCharts

Для успешного развертывания всех сервисов в Kubernetes кластере через ArgoCD необходимо подготовить и настроить Helm Charts для каждого сервиса.

Для каждого сервиса создайте отдельный ArgoCD Application, который будет управлять его развертыванием. Для всех сервисов настройте values.yaml, необходимо указать следующие параметры:

- Репозитории с образами: Укажите URL-адрес репозитория Docker образов для сервиса.

- PullSecret: Настройте секрет для доступа к приватным репозиториям с Docker образами. Укажите его в поле imagePullSecrets.

- Ингресс: Настройте доступ к сервисам через Ingress, используя заранее созданный домен. Укажите все необходимые правила и аннотации.

- StorageClass: Укажите класс хранилища для Persistent Volume, если сервис требует использования хранилища.

- Селекторы и толерации: Если требуется, укажите селекторы и толерации для подов, чтобы они размещались на узлах с определёнными характеристиками.

- Настройка секретов: В values-файлах укажите ссылки на существующие Kubernetes секреты, которые содержат критические данные для сервисов, такие как логины и пароли. Пример:

```
env:
 secret:
   postgresql-creds-secret:
     - DATASOURCE_USERNAME
     - DATASOURCE_URL
     - DATASOURCE_PASSWORD
     - DATASOURCE_SCHEMA
   rabbitmq-creds-secret:
     - RABBITMQ_URI
     - RABBITMQ_USERNAME
     - RABBITMQ_PASSWORD
```

Порядок установки сервисов

- Meilisearch: укажите название секрета backend-secret для параметра existingMasterKeySecret:

```
    auth:
      # Use an existing Kubernetes secret for the MEILI_MASTER_KEY
      existingMasterKeySecret: "backend-secret"
```

- STUNner Gateway Operator:

1. После деплоя applications в Арго создайте CRD для Gateway, GatewayClass, GatewayConfig и UDPRoute.

2. Пример конфигурации CRD:

Gateway:

```
apiVersion: gateway.networking.k8s.io/v1beta1
kind: Gateway
metadata:
  annotations:
    stunner.l7mp.io/service-type: LoadBalancer
  name: udp-gateway
  namespace: yuchat
spec:
  addresses:
    - type: IPAddress
      value: <Ваш IP>
  gatewayClassName: stunner-gatewayclass
  listeners:
    - allowedRoutes:
        namespaces:
          from: Same
      name: udp-listener
      port: 3478
      protocol: UDP
```

GatewayClass:

```
apiVersion: gateway.networking.k8s.io/v1beta1
kind: GatewayClass
metadata:
  name: stunner-gatewayclass
spec:
  parametersRef:
    group: stunner.l7mp.io
    kind: GatewayConfig
    name: stunner-gatewayconfig
    namespace: yuchat
  controllerName: "stunner.l7mp.io/gateway-operator"
```

GatewayConfig:

```
apiVersion: stunner.l7mp.io/v1alpha1
kind: GatewayConfig
metadata:
  name: stunner-gatewayconfig
  namespace: yuchat
spec:
  authType: plaintext
  password: <Ваш пароль>
  realm: stunner.l7mp.io
  stunnerConfig: stunnerd-config
  userName: <Ваш логин>
  logLevel: all:TRACE
```

UDPRoute:

```
kind: UDPRoute
metadata:
  name: media-yuchat
  namespace: yuchat
spec:
  parentRefs:
    - group: gateway.networking.k8s.io
      kind: Gateway
      name: udp-gateway
  rules:
    - backendRefs:
        - group: ""
          kind: Service
          name: webrtc
          namespace: yuchat
          weight: 1
          port: 8090
```

Остальные сервисы: Установите оставшиеся сервисы в следующем порядке:

1. web
2. nginx-static (за уточнением конфигурации обратитесь к разработчикам)
3. avnotes-streaming
4. backend: при необходимости создайте вручную ConfigMap с конфигурацией и укажите его в values.yaml:

```
configurationExistingCM: "backend-configuration"
```

За уточнением конфигурации и получением секрета с api ключами (backend-apikeys) обратитесь к разработчикам

5. ws-endpoint-service
6. conference-service-discovery
7. chat-message-event-service
8. jira-chat-bot
9. webrtc-fs-stack
10. media-composer
11. media-converter