

**Информация, необходимая для установки программного обеспечения
«Информационная платформа «Pulsar-APPS+»**

Содержание

1.	Установка Платформы.....	3
1.1.	Общие сведения о Платформе	3
1.1.1.	Требования к администратору.....	3
1.1.2.	Аппаратные требования к серверам Платформы.....	3
1.1.3.	Требования к ПО и сетевым доступам.....	5
1.1.4.	Логическая схема работы.....	7
1.1.5.	Сетевая связность	8
1.2.	Подготовка к запуску APPS.....	10
1.2.1.	Установка Docker	10
1.2.2.	Примеры установки Docker на разные системы.....	11
1.3.	Установка.....	14
1.3.1.	Общие сведения об установке	14
1.3.2.	Подключение сетевого каталога	14
1.3.3.	Установка APPS	16
1.4.	Запуск APPS.....	17
1.4.1.	Общие сведения.....	17
1.4.2.	Запуск в режиме одного сервера.....	18
2.	Получение доступа к функциям Платформы.....	22

1. Установка Платформы

1.1. Общие сведения о Платформе

1.1.1. Требования к администратору

Основные требования

Для полноценной установки, обновления и администрирования программного обеспечения «Информационная платформа «Pulsar-APPS+» (далее – Платформа, APPS), а также сопутствующих инфраструктурных компонентов, сотрудник должен обладать опытом работы со следующими компонентами и ПО Linux:

- Bash – для написания скриптов автоматизации и выполнения команд в оболочке;
- SSH – для безопасного удалённого доступа к серверам;
- Docker – для работы с контейнерами приложений;
- Docker Compose – для оркестрации многоконтейнерных приложений;
- Консольные редакторы: Vim, Nano – для редактирования конфигурационных файлов в терминале;
- Базовое понимание сетевых технологий — для настройки и диагностики сетевых подключений.

Дополнительные компетенции

Наличие опыта работы с нижеперечисленными технологиями является желательным:

- Ansible – для автоматизации конфигурирования и управления инфраструктурой;
- PostgreSQL – для работы с реляционными базами данных;
- Redis – как хранилище кэшированных данных и очередей;
- Elasticsearch и стек ELK (Elasticsearch, Logstash, Kibana) – для централизованного сбора, анализа и визуализации логов;
- Kafka – для работы с распределёнными потоками сообщений;
- Высокая доступность (HA) – для обеспечения отказоустойчивости сервисов.

1.1.2. Аппаратные требования к серверам Платформы

Для успешной работы приложений Платформы необходимо соблюдать определенные аппаратные требования, которые зависят от выбранного режима развертывания: односерверный или кластерный. Ниже представлены рекомендуемые конфигурации для различных сценариев использования.

Запуск в режиме одного сервера

В односерверном режиме APPS и ELK разворачиваются на одном физическом или виртуальном сервере. Этот вариант подходит для небольших нагрузок, когда количество подключенных пользователей не превышает 200 человек.

Компонент	CPU (Core)	RAM	Диск	OS
APPS + ELK	10 ядер (частота от 2.2 GHz)	32 ГБ	500 ГБ	Debian, Ubuntu, Astra Linux, РЕД ОС

Примечание. Режим односерверной установки удобен для тестирования и для небольших рабочих сред, но не обеспечивает отказоустойчивость. При увеличении числа пользователей или нагрузки рекомендуется перейти на кластерную архитектуру.

Запуск в режиме кластера

Кластерная конфигурация обеспечивает высокую доступность и масштабируемость системы. Для отказоустойчивости требуется минимум три сервера APPS. Если ожидается высокая нагрузка или требуется повышенная надежность, рекомендуется вынести кластер Redis на отдельные серверы.

Требования к серверам APPS

Компонент	CPU (Core)	RAM	Диск	OS
Сервер APPS	6 ядер (частота от 2.2 GHz)	16 ГБ	100 ГБ	Debian, Ubuntu, Astra Linux, РЕД ОС

ВНИМАНИЕ! Минимальное количество серверов APPS для отказоустойчивости – три. Это гарантирует работоспособность системы даже при выходе из строя одного из узлов.

Вынос Redis на отдельные серверы

Для повышения производительности и отказоустойчивости рекомендуется вынести кластер Redis на отдельные серверы. Количество серверов Redis должно быть не менее трех и не более шести.

Компонент	CPU (Core)	RAM	Диск	OS
Сервер Redis	6 ядер (частота от 2.2 GHz)	16 ГБ	100 ГБ	Debian, Ubuntu, Astra Linux, РЕД ОС

Выделение Redis на отдельные серверы снижает нагрузку на основные серверы APPS и позволяет эффективнее использовать ресурсы системы.

Требования к серверу ELK

Для сбора, анализа и визуализации логов используется стек ELK, который может быть размещен либо на том же сервере, что и APPS (в односерверном режиме), либо на отдельном сервере (в кластерном режиме).

Компонент	CPU (Core)	RAM	Диск	OS
Сервер ELK	6 ядер (частота от 2.2 GHz)	16 ГБ	500 ГБ	Debian, Ubuntu, Astra Linux, РЕД ОС

ВНИМАНИЕ! Объем дискового пространства для сервера ELK зависит от объема собираемых логов и установленных политик хранения данных. Рекомендуется регулярно

мониторить использование дискового пространства и при необходимости увеличивать его размер.

Дополнительные рекомендации

- **Частота процессора:** Минимальная частота процессора должна составлять 2.2 GHz для обеспечения достаточной производительности.
- **Хранилище:** Используйте SSD-накопители для повышения скорости работы системы, особенно для баз данных и индексов ELK.
- **Масштабирование:** При росте нагрузки можно горизонтально масштабировать систему, добавляя новые серверы APPS или увеличивая мощность существующих узлов.
- **Резервное копирование:** Обеспечьте регулярное резервное копирование данных для предотвращения потери информации в случае сбоя.

1.1.3. Требования к ПО и сетевым доступам

Платформа представляет собой набор микросервисов, запускаемых в контейнерах. Для работы потребуется установить инструмент контейнеризации Docker и надстройку над докером Docker-compose – локальный оркестратор, который позволяет запускать множество контейнеров одновременно и маршрутизировать потоки данных между ними.

Установка должна выполняться от имени пользователя, являющегося администратором системы (имеющего права root).

Требования к сетевым доступам

Для получения docker-образов требуется доступ к адресам:

- cr.yandex
- hub.docker.com

Если установка производится сотрудниками ПМТ, требуются доступ к следующим ресурсам:

- FTP сервер Postmodern:
 - ftp.pmtech.ru
- Репозитории системы на примере Ubuntu (скачивание и установка зависимостей необходимых для установки Docker):
 - ru.archive.ubuntu.com
 - security.ubuntu.com
- Репозиторий Docker (скачивание и установка Docker):
 - download.docker.com
- Доступ к Github для установки Docker compose (скачивание и установка Docker compose):
 - github.com
- Реестр образов (Docker Registry) на Яндекс-облаке (для скачивания docker-образов с компонентами от ПМТ):
 - cr.yandex

Требования к ПО

Перечень необходимых пакетов для запуска Платформы:

Пакет	Описание
vim	Терминальный текстовый редактор
jq	JSON-конвертер
mc	Файловый менеджер с текстовым интерфейсом типа Norton Commander
curl	Служебная программа командной строки, позволяющая взаимодействовать с множеством различных серверов по множеству различных протоколов с синтаксисом URL
wget	Консольная программа для загрузки файлов по сети. Поддерживает протоколы HTTP, FTP и HTTPS, а также поддерживает работу через HTTP прокси-сервер
cifs-utils	Утилиты для монтирования сетевых файловых систем CIFS
ca-certificates	Пакет для обновления корневых сертификатов на Linux
gnupg	Утилита для шифрования информации и создания электронных цифровых подписей
sshd	Серверный компонент, который позволяет обеспечить безопасный удалённый доступ к системе с использованием протокола SSH (Secure Shell)
netcat	Утилита командной строки, которая читает и записывает данные через сетевые подключения, используя протоколы TCP или UDP
telnet	Утилита, которая позволяет установить соединение и интерактивный канал связи с любым портом удалённого устройства
ping	Утилита командной строки, используемая для проверки доступности хоста в сети
mtr	Инструмент сетевой диагностики командной строки, который объединяет функциональность программ traceroute и ping
nmap	Утилита с открытым исходным кодом для исследования сети и аудита безопасности
tcpdump	Утилита для перехвата и анализа сетевых пакетов в Linux
ansible	Система управления конфигурациями, написанная на языке программирования
Python	Применяется для автоматизации настройки и развёртывания программного обеспечения

1.1.4. Логическая схема работы

Схема работы APPS

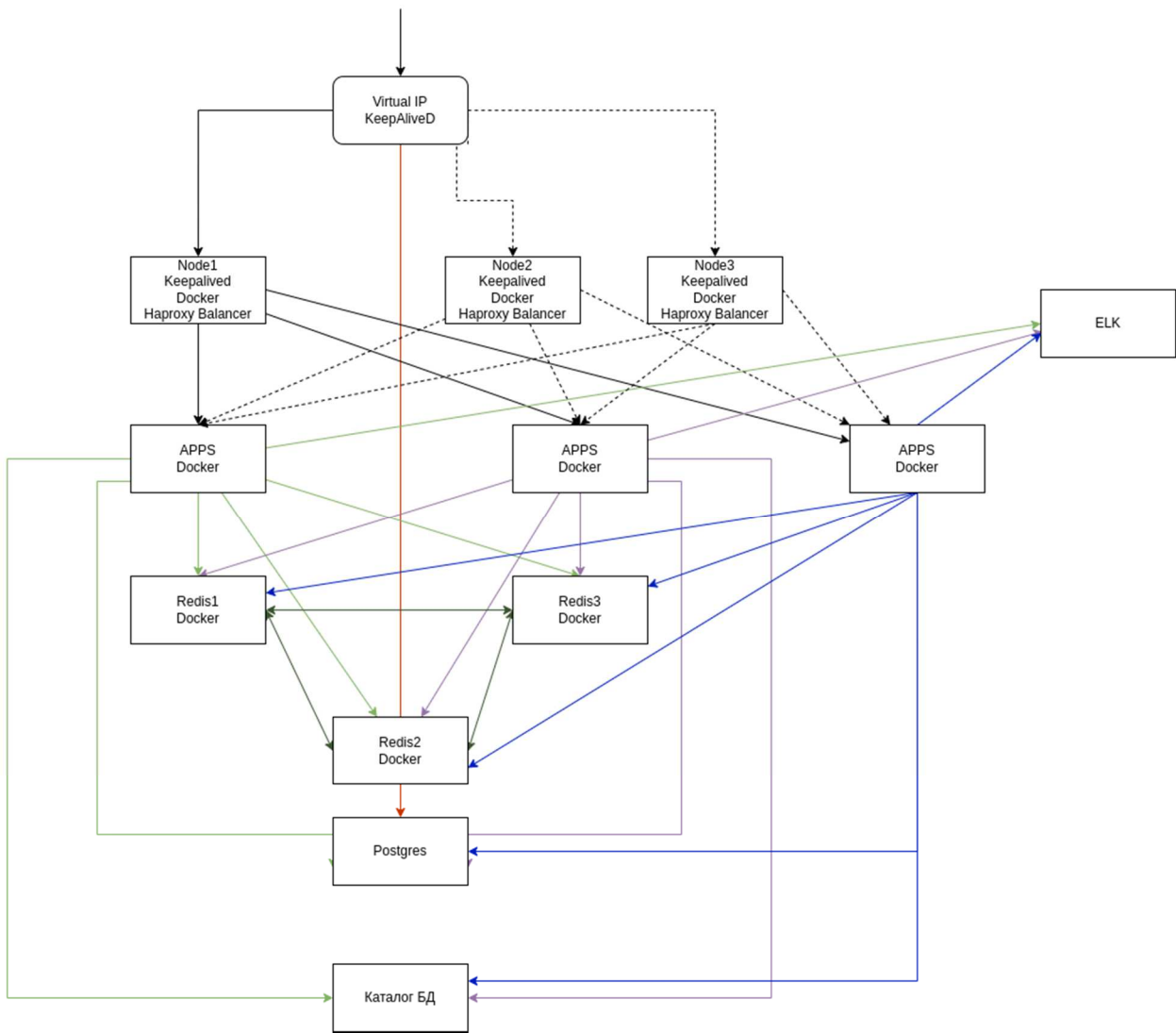


Рисунок 1. Схема работы APPS

Схема работы APPS в Docker

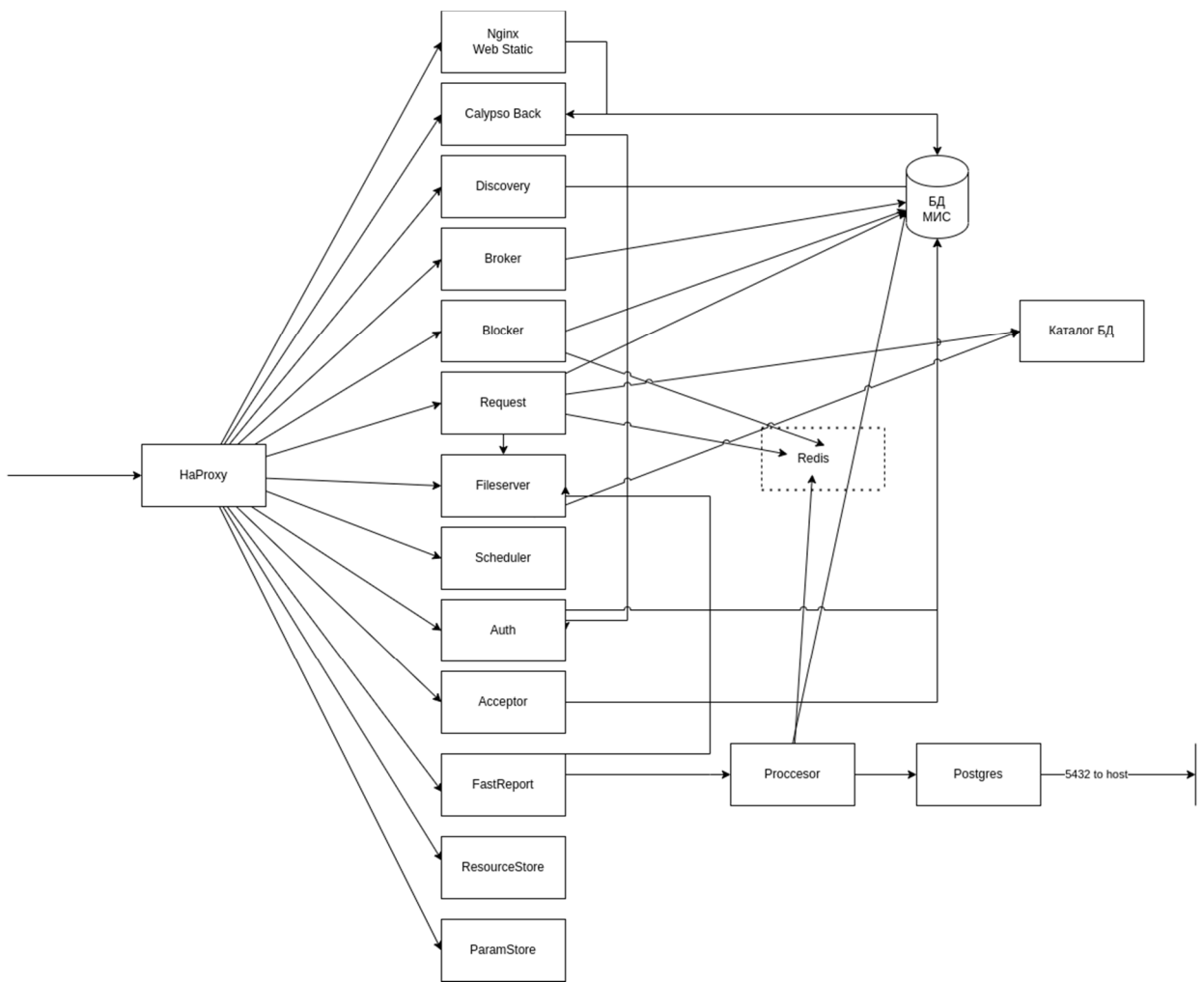


Рисунок 2. Схема работы APPS в Docker

1.1.5. Сетевая связность

Для работы платформы APPS поддерживаются два основных режима развертывания: односерверный и кластерный. Выбор режима зависит от требований к отказоустойчивости, масштабируемости и нагрузке на систему.

1. Режим одного сервера

В односерверном режиме все компоненты APPS, включая Redis, PostgreSQL и ELK, разворачиваются на одном физическом или виртуальном сервере. Этот вариант подходит для небольших организаций с ограниченным количеством пользователей и минимальными требованиями к отказоустойчивости.

Требования к сети

Ниже представлена таблица с описанием сетевых портов, используемых приложением APPS в односерверном режиме:

Приложение	Входящий порт	Исходящий адрес	Исходящий порт
APPS	18005/HTTP	Сервер MSSQL	1433/TCP

Приложение	Входящий порт	Исходящий адрес	Исходящий порт
APPS	18443/HTTPS	Сервер каталога базы	445/TCP
PostgreSQL	5432/TCP	—	—
Redis	6379/TCP	—	—
Filebeat	5048/TCP	Elasticsearch	9200/TCP
Kibana	5601/TCP	Elasticsearch	9200/TCP
APM-server	8200/TCP	Elasticsearch	9200/TCP

Примечание. Односерверный режим удобен для тестирования и для небольших рабочих сред, но не обеспечивает отказоустойчивость. При увеличении числа пользователей или нагрузки рекомендуется перейти на кластерную архитектуру.

2. Режим кластера

Кластерный режим развертывания APPS обеспечивает высокую доступность и масштабируемость системы. Для отказоустойчивости требуется минимум три сервера APPS, а также кластеры Redis и PostgreSQL (Patroni). Дополнительно потребуются два IP-адреса для кластерных целей:

- Один IP-адрес для балансировщика приложений HAProxy (APPS).
- Один IP-адрес для балансировщика PostgreSQL Patroni.

Требования к сети

Ниже представлена таблица с описанием сетевых портов, используемых приложением APPS в кластерном режиме:

Приложение	Входящий порт	Исходящий адрес	Исходящий порт
APPS	18005/HTTP	Сервер MSSQL	1433/TCP
	18443/HTTPS	Сервер каталога базы	445/TCP
		Postgres	5432/TCP
		Redis	6379/TCP
HAproxy balancer apps	8080/TCP	Apps (все ноды)	18005
	8405/TCP		
Postgres patroni	5432/TCP	Ноды Etc patroni	2379/TCP
		Ноды Etc patroni	2380/TCP
Etc patroni	2380/TCP	Ноды Etc patroni (синхронизация кластера)	2380/TCP
	2379/TCP		
HAproxy patroni	5000/TCP	Postgres patroni	5432/TCP
	5001/TCP		
Redis	6379/TCP		
Redis-sentinel	26379/TCP	Redis	6379/TCP
Filebeat	5048/TCP	Elasticsearch	9200/TCP
		Kibana	5432/TCP
Elasticsearch	6379/TCP		
Kibana	5601/TCP	Elasticsearch	9200/TCP
APM-server	8200/TCP	Elasticsearch	9200/TCP

1.2. Подготовка к запуску APPS

1.2.1. Установка Docker

Docker — это платформа с открытым исходным кодом для разработки, доставки и запуска приложений в изолированных контейнерах. Docker Compose — инструмент для определения и запуска многоконтейнерных приложений Docker с помощью YAML-файлов.

Docker поддерживается большинством современных операционных систем, включая Linux и Windows. Многие современные операционные системы уже включают Docker в свой состав.

Для установки Docker необходимо обратиться к документации вашей операционной системы.

Примечание. Установка должна выполняться от имени пользователя, являющегося администратором системы (имеющего права root).

В следующем разделе приведены примеры установки Docker на различные операционные системы.

Последние версии Docker

Если вам нужны последние версии Docker, рекомендуется использовать официальную инструкцию с сайта Docker для вашей операционной системы: [Install Docker](#).

Настройка пользователя для работы с Docker

После установки Docker необходимо добавить пользователя, от имени которого вы планируете запускать приложения, в группу docker. Это позволит избежать необходимости использования команды `sudo` каждый раз при работе с Docker.

Добавьте пользователя в группу docker с помощью следующей команды, заменив `имя_пользователя` на имя вашего пользователя:

```
sudo usermod -aG docker имя_пользователя
```

Перезапуск сессии

Для применения изменений необходимо начать новую сессию для данного пользователя.

Выполните выход из текущей сессии с помощью команды:

```
exit
```

Затем залогиньтесь снова.

Проверка принадлежности к группе docker

Чтобы проверить, что пользователь был успешно добавлен в группу docker, выполните команду:

```
id
```

В выводе команды должно быть указано, что пользователь принадлежит к группе docker. Пример вывода:

```
uid=1000(all) gid=1000(all) группы=1000(all),964(docker)
```

Если группа docker присутствует в списке, настройка завершена успешно.

1.2.2. Примеры установки Docker на разные системы

ВНИМАНИЕ! В данном разделе описаны примеры установки Docker и Docker Compose на популярные Linux-дистрибутивы, но если вам нужны последние версии Docker, рекомендуется использовать официальную инструкцию с сайта Docker для вашей операционной системы.

Установка Docker и Docker Compose

На Debian/Ubuntu

Подготовка системы:

```
$ sudo apt update && sudo apt upgrade -y
```

Установка необходимых пакетов:

```
$ sudo apt install -y apt-transport-https ca-certificates curl software-propertiescommon  
gnupg lsb-release
```

Добавление GPG-ключа Docker:

```
curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo gpg --dearmor -o  
/usr/share/keyrings/docker-archive-keyring.gpg
```

Добавление репозитория Docker:

```
echo "deb [arch=$(dpkg --print-architecture) signed-  
by=/usr/share/keyrings/dockerarchive-  
keyring.gpg] https://download.docker.com/linux/ubuntu $(lsb_release -cs)  
stable" | sudo tee /etc/apt/sources.list.d/docker.list > /dev/null
```

Установка Docker:

```
sudo apt update; sudo apt install -y docker-ce docker-ce-cli containerd.io
```

Установка Docker Compose:

```
sudo curl -L  
"https://github.com/docker/compose/releases/download/v2.20.2/dockercompose-$(  
uname -s)-$(uname -m)" -o /usr/local/bin/docker-compose; sudo chmod +x  
/usr/local/bin/docker-compose
```

Добавление текущего пользователя в группу docker:

```
sudo usermod -aG docker $USER ; newgrp docker
```

На RedOS

Подготовка системы:

```
sudo yum update -y
```

Установка Docker:

```
sudo yum install -y yum-utils device-mapper-persistent-data lvm2
```

```
sudo yum-config-manager --add-repo  
https://download.docker.com/linux/centos/dockerce.
```

repo

```
sudo yum install -y docker-ce docker-ce-cli containerd.io
```

Запуск службы Docker:

```
sudo systemctl start docker
```

```
sudo systemctl enable docker
```

Установка Docker Compose:

```
sudo curl -L  
"https://github.com/docker/compose/releases/download/v2.20.2/dockercompose-$(  
uname -s)-$(uname -m)" -o /usr/local/bin/docker-compose  
sudo chmod +x /usr/local/bin/docker-compose
```

Добавление текущего пользователя в группу docker:

```
sudo usermod -aG docker $USER
```

```
newgrp docker
```

На Astra Linux

ВНИМАНИЕ! В Astra Linux рекомендуется использовать официальные репозитории дистрибутива.

Подготовка системы:

```
sudo apt update && sudo apt upgrade -y
```

Установка Docker:

```
sudo apt install -y docker.io
```

Запуск службы Docker:

```
sudo systemctl start docker
```

```
sudo systemctl enable docker
```

Установка Docker Compose:

```
sudo curl -L "https://github.com/docker/compose/releases/download/v2.20.2/dockercompose-$(uname -s)-$(uname -m)" -o /usr/local/bin/docker-compose
sudo chmod +x /usr/local/bin/docker-compose
```

Добавление текущего пользователя в группу docker:

```
sudo usermod -aG docker $USER
newgrp docker
```

На ALT Linux

Подготовка системы:

```
sudo apt-get update -a
```

Установка Docker:

```
sudo apt-get docker-engine
```

Запуск службы Docker:

```
sudo systemctl start docker
```

```
sudo systemctl enable docker
```

Установка Docker Compose:

```
sudo curl -L "https://github.com/docker/compose/releases/download/v2.20.2/dockercompose-$(uname -s)-$(uname -m)" -o /usr/local/bin/docker-compose
sudo chmod +x /usr/local/bin/docker-compose
```

Добавление текущего пользователя в группу docker:

```
sudo groupadd docker
sudo usermod -aG docker $USER
newgrp docker
```

Проверка установки

Для проверки успешности установки выполните следующие команды:

```
docker --version
```

```
docker-compose --version
```

Если версии отображаются корректно, установка прошла успешно.

1.3. Установка

1.3.1. Общие сведения об установке

Установить APPS можно двумя способами – online и offline.

Установка Online

Режим установки online требует наличие возможности скачать образы docker-контейнеров из сети интернет с адресов указанных в разделе 1.1.3 («Требования к ПО и сетевым доступам»).

Установка Offline

Режим установки offline не требует доступа к сети интернет, всё что нужно для работы уже заложено в инсталлятор. Данный режим обеспечивает высокую безопасность, но менее гибок, поскольку инсталлятор имеет только одну определенную версию. В этом случае отсутствует гибкость обновления и отката, нет высокой скорости загрузки.

1.3.2. Подключение сетевого каталога

Для работы службы `fileservice` требуется подключить каталог базы данных к серверу(ам) с APPS.

Простое подключение каталога можно произвести командой `mount` с передачей параметров `mount //имя_сервер/папка /точка/монтирования -o параметры`.

ВНИМАНИЕ! Требуется предварительно установленного пакета `cifs-utils`.

Пример команды:

```
mount //server/share /mnt -o username=user
```

Данный вариант подключения подходит для разовых задач, так как не поддерживает автоматического переподключения в случае обрыва сети и других проблем подключения.

Для обхода данной проблемы следует использовать механизм монтирования с помощью **systemd**.

Создайте файл, имя которого содержит в себе путь, куда будет монтироваться сетевой каталог. Для примера (и по умолчанию) используется каталог `/mnt` для которого файл будет называться `mnt.mount`. Первый слэш игнорируется, все последующие заменяются дефисом (пример более длинного пути: `/mnt/share - mnt-share.mount`):

```
sudo touch /etc/systemd/system/mnt.mount
```

Откройте файл на редактирование:

```
sudo vim /etc/systemd/system/mnt.mount
```

Перейдите в режим редактирования нажатием клавиши `i` и вставьте следующее содержимое:

```
[Unit]
```

Description=cifs mount script

Requires=network-online.target

After=network-online.service

[Mount]

What=//windows/share

Where=/mnt

Options=username=user,password=password,workgroup=domain_name,icharset=utf8,rw,file_m

ode=0777,dir_mode=0777,vers=2.0,rw

Type=cifs

[Install]

WantedBy=multi-user.target

Измените адрес расшаренной папки, имя пользователя, пароль и имя домена на ваши и сохраните командами:

ESC : wq enter

По аналогии создайте файл для автомонтирования:

sudo touch /etc/systemd/system/mnt.automount

Откройте файл на редактирование:

sudo vim /etc/systemd/system/mnt.automount

Перейдите в режим редактирования нажатием клавиши *i* и вставьте следующее содержимое:

[Unit]

Description=cifs mount script

Requires=network-online.target

After=network-online.service

[Automount]

Where=/mnt

TimeoutIdleSec=10

[Install]

WantedBy=multi-user.target

Сохраните файл. Если вы не изменяли путь монтирования */mnt*, то никакие изменения не требуются.

ESC : wq enter

Теперь необходимо применить системные настройки командой:

```
sudo systemctl daemon-reload
```

Включите автозагрузку созданных настроек:

```
sudo systemctl enable mnt.mount
```

```
sudo systemctl enable mnt.automount
```

Запустите созданные настройки:

```
sudo systemctl start mnt.mount
```

```
sudo systemctl start mnt.automount
```

Для проверки нужно обратиться к каталогу */mnt* командой:

```
ls /mnt
```

Если все настроено правильно, вы увидите листинг файлов из каталога.

1.3.3. Установка APPS

Установка стека APPS для кластерного режима и режима одного сервера не отличается, отличия будут отражены в разделе «Запуск APPS». В целом, после установки Docker и проведения необходимых предварительных настроек, система готова к установке APPS.

Получение токена доступа

Для начала установки APPS необходимо запросить токен доступа к реестру образов Docker у технической поддержки. Токен действителен в течение нескольких часов.

Пример токена:

```
t1.9euelZrJio-KzprOjpmJlJeLzYrPkO3rnpWakJ2akc6Jy5iXz42Vy5yWx5Ll9Pd-  
YUZEe9bQyzL3fT3PhBERPnvW0Msy83n9euelZqMls6YjZfKkl_PnpeSzcyWy-  
_8zef1656VmsqOl5KQypianZePlJTHnluY7_3F656VmoyWzpiNl8qQj8-  
el5LNzJbL.ns5AqMcTXsg4nVVpOxaDwHhmxA8lBYBqFPO8aG1CLwGrtttFruH476u6QnfhvU4  
YgDubseVcxwTi9g05kFNWBg
```

Аутентификация в реестре Docker

После получения токена выполните аутентификацию в реестре Docker командой, заменив *<token>* на ваш токен:

```
docker login cr.yandex -u iam -p <token>
```

Скачивание контейнера для установки APPS

Скачайте docker-контейнер для установки APPS с помощью следующей команды:

```
docker pull cr.yandex/crp65nr1nr7bt8d4gb3d/init-apps:latest
```

Запуск процесса установки

По умолчанию APPS устанавливается в каталог /opt/PMT/APPS. Запустите процесс установки командой:

```
docker run --rm -v /opt/PMT:/opt/PMT cr.yandex/crp65nr1nr7bt8d4gb3d/init-apps:latest
```

Настройка прав доступа

После завершения установки необходимо исправить права доступа для созданного каталога. Выполните следующую команду:

```
sudo chown ${USER}:${USER} /opt/PMT -R
```

Теперь установка APPS успешно завершена.

1.4. Запуск APPS

1.4.1. Общие сведения

Режим одного сервера (Standalone Mode)

Описание

Это самый простой режим работы системы — вся нагрузка обрабатывается одним узлом (сервером). Все данные хранятся и обрабатываются локально, без взаимодействия с другими узлами.

Преимущества:

- простота настройки
- минимальные требования к ресурсам
- легкость администрирования и отладки
- подходит для тестирования и небольших проектов

Недостатки:

- отсутствие отказоустойчивости: если сервер выйдет из строя, сервис станет недоступным
- ограниченная производительность
- отсутствие масштабируемости

Кластерный режим (Cluster Mode)

Описание

Кластер — это группа серверов, работающих вместе как единая система. Режим позволяет распределять нагрузку, обеспечивать отказоустойчивость и горизонтальное масштабирование.

Преимущества:

- высокая доступность (HA)
- балансировка нагрузки между узлами
- возможность масштабирования по мере роста нагрузки
- распределение данных и обработки задач
- защита от потери данных (при правильной настройке)

Недостатки:

- сложность настройки и управления
- увеличенные аппаратные/ресурсные затраты
- необходимость синхронизации между узлами
- более сложная диагностика и отладка

1.4.2. Запуск в режиме одного сервера

В данном разделе рассматривается вариант запуска на одном сервере, в качестве примера взят адрес *192.168.10.1* (при настройке во всех разделах его необходимо заменить на свой).

Предварительные требования:

- Установлен APPS.
- Выбран режим логирования.

Примечание. APPS умеет логировать в файлы, но мы рекомендуем использовать систему ELK для записи логов и трассировок.

Для начала настройки перейдите в каталог */opt/PMT/APPS*, в который предварительно был установлен APPS:

```
cd /opt/PMT/APPS/
```

Настройка и конфигурация среды запуска производится в файле **.env**.

На текущий момент файл **.env** содержит переменные, позволяющие по максимуму синхронно настроить все сервисы, с минимальными правками в других файлах.

В начале файла **.env** перечислены образы и теги образов из которых запускается APPS. Теги образов являются версией компонентов, которые используются в стеке APPS. Запросить текущие актуальные версии можно в технической поддержке support@pmttech.ru.

В блоке *local scaling* можно увеличить количество контейнеров, работающих под нагрузкой, данный блок требует понимания архитектуры стека APPS.

По умолчанию APPS настроен на порт 18005 и использование внутреннего Redis и Postgres. Откройте файл **.env** и проверьте следующие переменные:

Переменная	Описание
port_web=18005	Порт, который использует APPS
COMPOSE_PROFILES=db	По умолчанию используется встроенный redis и postgres – отключить(убрать db). Если требуется использовать внешние, необходимо внести изменения в файл csb.env .

	Далее в данной настройке будут использоваться профили различных компонентов стека APPS, которые можно подключать и отключать.
--	---

В блоке Redis conf настраивается версия и прочие настройки Redis, которая используется внутри стека APPS.

Переменная	Описание
redis_version=7.4	Версия Redis
port_redis=6379	Порт Redis
redis_pw=Str0ngP@ssw0rd	Пароль Redis

В блоке Pg conf настраивается версия и прочие настройки Postgres, которая используется внутри стека APPS.

Переменная	Описание
pg_version=17.5-alpine	Версия образа Postgres
pg_user=sa	Пользователь Postgres
pg_password=password	Пароль Postgres

В блоке common settings настраиваются общие параметры для других сервисов

Переменная	Описание
workspace=APPS	Переменная указывающая окружение. Используется при кластеризации
ext_ip=192.168.88.22	Внешний IP сервера

В файле csb.env содержатся основные настройки стека APPS. Откройте его на редактирование и измените следующие переменные:

Переменная	Описание
SQL_SERVER=MSSQLSERVERInstance	Имя сервера с базой данных default ¹
DATABASE=name_of_database	Имя базы данных default
SQL_USER=user	Пользователь для подключения к серверу MSSQL с базой данных default
SQL_PASSWORD=password	Пользователь для подключения к серверу MSSQL с базой данных default
POSTGRES_CONNECTION=postgres://\$pg_user:\$pg_password@postgres:5432/postgres	По умолчанию используется встроенный Postgres, генерится на основе переменных из файла .env
EXTERNAL_URL=http://\$ext_ip:\$port_web/%service%	Внешний адрес сервиса APPS, генерится на основе переменных из файла .env
LOG_TYPE=remote	Тип логирования remote/file. Для ELK используем remote, для логирования в файлы – file. Настоятельно рекомендуем использовать ELK, в противном случае траблшутинг будет практически невозможен
LOG_HOST=192.168.10.4	Ip filebeat (если он используется)
LOG_PORT=5048	Порт filebeat (если он используется)
LOG_USERNAME=	Логин Kibana
LOG_PASSWORD=	Пароль Kibana
TELEMETRY_ACTIVE=true	Включение сервера трассировок

¹ Опционально поддерживается возможность подключения к БД под управлением MS-SQL

Переменная	Описание
TELEMETRY_SERVER_URL=http://192.168.10.1:8200/	Адрес сервиса APM
TELEMETRY_PANEL_URL=http://192.168.10.1:5601/	Адрес Kibana
TELEMETRY_ENVIRONMENT=apps1	Параметр, показывающий, с какого сервера переданы логи. Задается автоматически на основании файла .env

Файл `common.env` содержит общий конфигурационный блок, передающийся всем службам, кроме `discovery`. Отредактируйте в нем следующие переменные:

Переменная	Описание
EXTERNAL_URL=http://\$ext_ip:\$port_web/%service%	Внешний адрес сервиса APPS, формируется на основании переменных из файла .env
TZ=Europe/Moscow	Часовой пояс (нужно изменить при необходимости)
LINUX_ROOT="/mnt"	Точка монтирования каталога для хранения файлов в Linux. Изменение данного значения требует понимания работы стека APPS.
WINDOWS_ROOT="\\\\192.168.10.10\\shared_folder"	Сетевой путь к каталогу для хранения файлов (необходимо указывать с символами экранирования)

Файл `frs.env` отвечает за настройки сервиса FastReport. Отредактируйте в нем следующие переменные:

Переменная	Описание
EXTERNAL_URL=http://\$ext_ip:\$port_web/fastreport	Внешний адрес сервиса в APPS, формируется на основании переменных из файла .env

Запустите APPS, выполнив следующую команду из текущей директории:

```
docker-compose up -d
```

Проверьте, что APPS работает:

```
docker-compose ps
```

Все сервисы должны быть в состоянии Up, как в данном примере (названия сервисов могут отличаться от приведенных в примере):

```
NAME IMAGE COMMAND SERVICE CREATED STATUS PORTS
```

```
testers_800_dev_btk-acceptor-1 cr.yandex/crp65nr1nr7bt8d4gb3d/csb2.test:latest
```

```
"docker-entrypoint.s..." acceptor 9 days ago Up 3 days
```

```
testers_800_dev_btk-auth-1 cr.yandex/crp65nr1nr7bt8d4gb3d/csb2.test:latest
```

```
"docker-entrypoint.s..." auth 9 days ago Up 3 days
```

```
testers_800_dev_btk-blocker-1 cr.yandex/crp65nr1nr7bt8d4gb3d/csb2.test:latest
```

```
"docker-entrypoint.s..." blocker 9 days ago Up 3 days
```

```
testers_800_dev_btk-broker-1 cr.yandex/crp65nr1nr7bt8d4gb3d/csb2.test:latest
```

```
"docker-entrypoint.s..." broker 9 days ago Up 3 days
```

```
testers_800_dev_btk-cacheset-1 cr.yandex/crp65nr1nr7bt8d4gb3d/csb2.test:latest
```

```
"docker-entrypoint.s..." cacheset 9 days ago Up 3 days
```

testers_800_dev_btk-calypso_back-1
cr.yandex/crpb82v5uct0slolk11a/calypso_back.test:latest "dockerentrypoint.s..." calypso_back 9 days ago Up 3 days
testers_800_dev_btk-discovery-1 cr.yandex/crp65nr1nr7bt8d4gb3d/csb2.test:latest "docker-entrypoint.s..." discovery 9 days ago Up 3 days (healthy)
testers_800_dev_btk-fastreport-1 cr.yandex/crp65nr1nr7bt8d4gb3d/frs.test:latest "dotnet PmtFastRepor..." fastreport 9 days ago Up 3 days (healthy)
testers_800_dev_btk-fileserver-1 cr.yandex/crp65nr1nr7bt8d4gb3d/csb2.test:latest "docker-entrypoint.s..." fileserver 9 days ago Up 3 days (healthy)
testers_800_dev_btk-haproxy-1 cr.yandex/crp65nr1nr7bt8d4gb3d/haproxy:latest "/docker-entrypoint...." haproxy 9 days ago Up 3 days
0.0.0.0:20443->443/tcp, :::20443->443/tcp, 0.0.0.0:20005->8080/tcp, :::20005->8080/tcp, 0.0.0.0:8444->8404/tcp, :::8444->8404/tcp
testers_800_dev_btk-info-1 cr.yandex/crp65nr1nr7bt8d4gb3d/csb2.test:latest "docker-entrypoint.s..." info 9 days ago Up 3 days
testers_800_dev_btk-onlinequeue-1 cr.yandex/crp65nr1nr7bt8d4gb3d/csb2.test:latest "docker-entrypoint.s..." onlinequeue 9 days ago Up 3 days
testers_800_dev_btk-paramstore-1 cr.yandex/crp65nr1nr7bt8d4gb3d/csb2.test:latest "docker-entrypoint.s..." paramstore 9 days ago Up 3 days
testers_800_dev_btk-powermeter-1 cr.yandex/crp65nr1nr7bt8d4gb3d/csb2.test:latest "docker-entrypoint.s..." powermeter 9 days ago Up 3 days
testers_800_dev_btk-processor-1 cr.yandex/crp65nr1nr7bt8d4gb3d/csb2.test:latest "docker-entrypoint.s..." processor 9 days ago Up 3 days
testers_800_dev_btk-processor-2 cr.yandex/crp65nr1nr7bt8d4gb3d/csb2.test:latest "docker-entrypoint.s..." processor 9 days ago Up 3 days
testers_800_dev_btk-processor-3 cr.yandex/crp65nr1nr7bt8d4gb3d/csb2.test:latest "docker-entrypoint.s..." processor 9 days ago Up 3 days
testers_800_dev_btk-request-1 cr.yandex/crp65nr1nr7bt8d4gb3d/csb2.test:latest "docker-entrypoint.s..." request 9 days ago Up 3 days (healthy)
testers_800_dev_btk-resourcestore-1 cr.yandex/crp65nr1nr7bt8d4gb3d/csb2.test:latest "docker-entrypoint.s..." resourcestore 9 days ago Up 3 days
testers_800_dev_btk-rtfconverter-1

```
cr.yandex/crpb82v5uct0slolk11a/rtfconverter.test:latest "dotnet
RtfConverter..." rtfconverter 9 days ago Up 3 days
0.0.0.0:3015->3015/tcp, :::3015->3015/tcp
testers_800_dev_btk-scheduler-1 cr.yandex/crpb65nr1nr7bt8d4gb3d/csb2.test:latest
"docker-entrypoint.s..." scheduler 9 days ago Up 3 days
testers_800_dev_btk-webserver-1
cr.yandex/crpb65nr1nr7bt8d4gb3d/apps_common_nginx2.test:latest "/dockerentrypoint. ..."
webserver 9 days ago Up 3 days 80/tcp
testers_800_dev_btk-xsltconverter-1
cr.yandex/crpb82v5uct0slolk11a/xsltconverter.main:latest "node
build/index.js..." xsltconverter 9 days ago Up 3 days
```

Эта проверка показывает, что сервисы, как инфраструктурные единицы, работают, теперь необходимо проверить логику работы.

Первым этапом надо проверить работу сервиса `discovery`, для этого нужно обратиться к нему с помощью команды `curl` по его реальному адресу (здесь адрес из примера). Ответ сервиса может появляться не сразу после старта, сервису некоторое время на инициализацию:

```
curl http://192.168.10.1:18005/discovery
```

В ответ вы должны получить:

```
discovery service is running
```

Это означает, что сервис запущен.

Далее требуется проверить, что сервис содержит рабочие данные:

```
curl http://192.168.10.1:18005/discovery/discover|jq|more
```

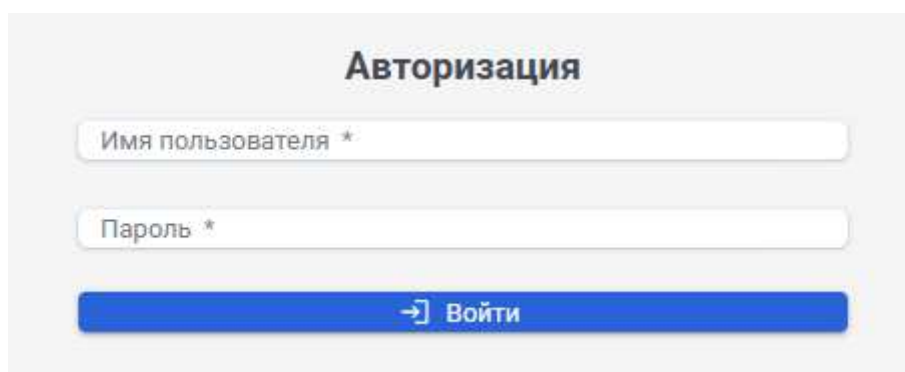
На выходе должны быть данные в JSON с настройками сервисов.

Последняя проверка – это обращение по адресу `http://192.168.10.1:18005` из браузера. Вы должны увидеть окно логина (см. Рисунок 3). На данном этапе настройка APPS завершена.

2. Получение доступа к функциям Платформы

Для получения доступа к функциям Платформы необходимо открыть браузер и пройти авторизацию на сайте, определенном при развертывании Платформы:

1) Проверка возможности входа в программу осуществляется через веб-браузер:



The image shows a login window titled "Авторизация". It contains two input fields: "Имя пользователя *" and "Пароль *". Below these fields is a blue button with a right-pointing arrow and the text "Войти".

Рисунок 3. Окно логина

2) Вход в систему осуществляется по Имени пользователя (логину) и Пароллю.

Для полноценного использования функций ПО «Информационная платформа «Pulsar-APPS+» рабочее место должно быть оборудовано доступом в интернет со скоростью не менее 1 Мбит/с, а также браузером на базе ядра Chromium версии не ниже 121.