Правообладатель: Общество с ограниченной ответственностью «Интернет для жизни»
(ООО «Интернет для жизни»)
430005, Республика Мордовия, г. Саранск, ул. Большевистская, д. 11, офис 201,
ОГРН 1081326002724, ИНН 1326207059

ПЛАТФОРМА ОБРАБОТКИ ДАННЫХ НА ОСНОВЕ ИСКУССТВЕННОГО ИНТЕЛЛЕКТА

Руководство по техническому обслуживанию

RU.86213171.61897-01 46

Листов 20

2022

АННОТАЦИЯ

Настоящий документ содержит сведения, необходимые для эксплуатации программы «Платформа обработки данных на основе искусственного интеллекта», предназначенной для автоматизации процессов анализа и детектирования по фотограмметрическим данным объектов, с определением их типа, занимаемой площади и физического расположения.

Оформление программного документа «Руководство по техническому обслуживанию» произведено по требованиям ЕСПД (ГОСТ 19.101-77[1], ГОСТ 19.103-77[2], ГОСТ 19.104-78[3], ГОСТ 19.105-78[4], ГОСТ 19.508-79[5]).

[1] ГОСТ 19.101-77 ЕСПД Виды программ и программных документов
[2] ГОСТ 19.103-77 ЕСПД Обозначение программ и программных документов
[3] ГОСТ 19.104-78 ЕСПД Основные надписи
[4] ГОСТ 19.105-78 ЕСПД Общие требования к программным документам
[5] ГОСТ 19.508-79 ЕСПД Руководство по техническому обслуживанию Требования к содержанию и оформлению

СОДЕРЖАНИЕ

1. ВВЕДЕНИЕ

Данный документ представляет собой руководство по техническому обслуживанию программы «Платформа обработки данных на основе искусственного интеллекта» (далее – Платформа).

Документ предназначен специалисту по техническому обслуживанию и администратору Платформы на этапе сопровождения.

При осуществлении работ также следует руководствоваться следующими документами:

- RU.86213171.61897-01 13 Описание программы
- RU.86213171.61897-01 31 Описание применения
- RU.86213171.61897-01 32 Руководство системного программиста
- RU.86213171.61897-01 33 Руководство программиста
- RU.86213171.61897-01 34 Руководство оператора

2. ОБЩИЕ УКАЗАНИЯ

2.1. Порядок технического обслуживания

В рамках сопровождения Платформы специалистом по техническому обслуживанию реализуется выполнение следующих функций:

- Обновление программного обеспечения;
- Выполнение резервного копирования;
- Осуществление проверки штатными средствами работоспособности Платформы;
- Восстановление работоспособности Платформы при аварийных ситуациях.

Требования к уровню подготовки специалиста по техническому обслуживанию приведены в п.2.2. настоящего программного документа.

2.2. Требования к персоналу

Выполнение функций, указанных в п.2.1. данного программного документа, предполагает соответствие специалиста по техническому обслуживанию следующим требованиям:

- Системное администрирование Linux;
- Администрирование СУБД MongoDB, PostgreSQL;
- Администрирование систем контейнеризации Docker, кластеризации Docker Swarm;
- Навыки работы с системой мониторинга Prometheus и системы визуализации Grafana.

3. ТРЕБОВАНИЯ К ТЕХНИЧЕСКИМ СРЕДСТВАМ

3.1. Минимальный состав технических средств программы

Рекомендуемые требования к аппаратной части, на которой планируется развёртывание и функционирование Платформы:

- Процессор: 16x CPU 3.5 GHz.
- Оперативная память: 64 Gb.
- Видео карта: NVIDIA GeForce RTX 3080, 10 Gb.
- SSD накопитель: 1 Tb SSD.
- Жесткий диск: 18 Tb HDD.
- Сетевой адаптер: Gigabit Ethernet 1000 Mbit.

4. ОПИСАНИЕ ФУНКЦИЙ

4.1. Обновление программного обеспечения

В связи с тем, что Платформа построена на микросервисной

архитектуре, где в качестве среды контейнеризации используется Docker и для отказоустойчивости серверы объединены в кластер Docker Swarm, то под обновлением программного обеспечения Платформы понимается обновление образов Docker (обновление образом Docker программных микросервисов Платформы), из которых разворачиваются контейнеры.

Для обновления микросервиса новым образом нужно:

1. Подключиться к любому серверу кластера Docker Swarm по протоколу ssh от имени пользователя, обладающего правами суперпользователя:

ssh someuser@server-ip-address – из среды Linux,

либо с помощью программы Putty из среды Windows.

2. Выполнить поиск микросервиса, который нужно обновить, по его имени. Например:

sudo docker service ls |grep kpp

Примерный вывод команды:

*x5js3eqe4j9j aip_sier-kppreplicated 1/1 newharbor.evolenta.ru/sier/kpp:v45-mysql *:81->80/tcp, *:9192->9192/tcp*

3. Обновить образ, используя имя микросервиса, найденное в п.2. Например, для обновления микросервиса *aip_sier-kpp* нужно выполнить:

docker service update --with-registry-auth --image newharbor.evolenta.ru/sier/kpp:v46-mysql aip_sier-kpp,

где ключ (image) – позволяет ввести имя нового образа.

4. Проверить состояние микросервиса командой:

sudo docker service ls |grep aip_sier-kpp

Примерный вывод команды:

*x5js3eqe4j9j aip_sier-kpp replicated 1/1 newharbor.evolenta.ru/sier/kpp:v46-mysql *:81->80/tcp, *:9192->9192/tcp*

Убедиться, что количество запущенных и запланированных к запуску реплик микросервиса одинаково (*replicated 1/1*).

Если управление микросервисами осуществляется с помощью графической оболочки программного обеспечения «Portainer» (устанавливается отдельно), то для обновления необходимо:

1. Авторизоваться в программном обеспечении «Portainer» с правами администратора по адресу:

http://ip-address:9000,

где ip-address – ip-адрес или доменное имя, на котором развернуты микросервисы.

2. Открыть пункт меню «Services» и выбираем микросервис, который требуется обновить, нажав на имя левой кнопкой мыши. При необходимости осуществить поиск микросервиса по его наименованию, заполнив поле «Search» (Рисунок 1).

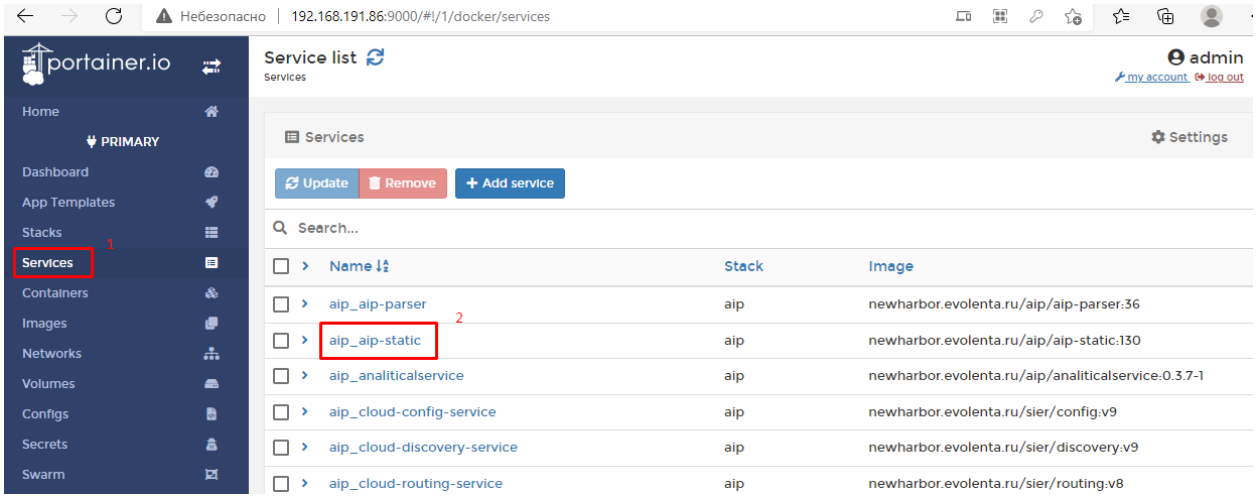

Рисунок 1 – Интерфейс пункта меню «Services» в программном обеспечении Portainer

3. Выбрать пункт меню «Container image» (Рисунок 2).

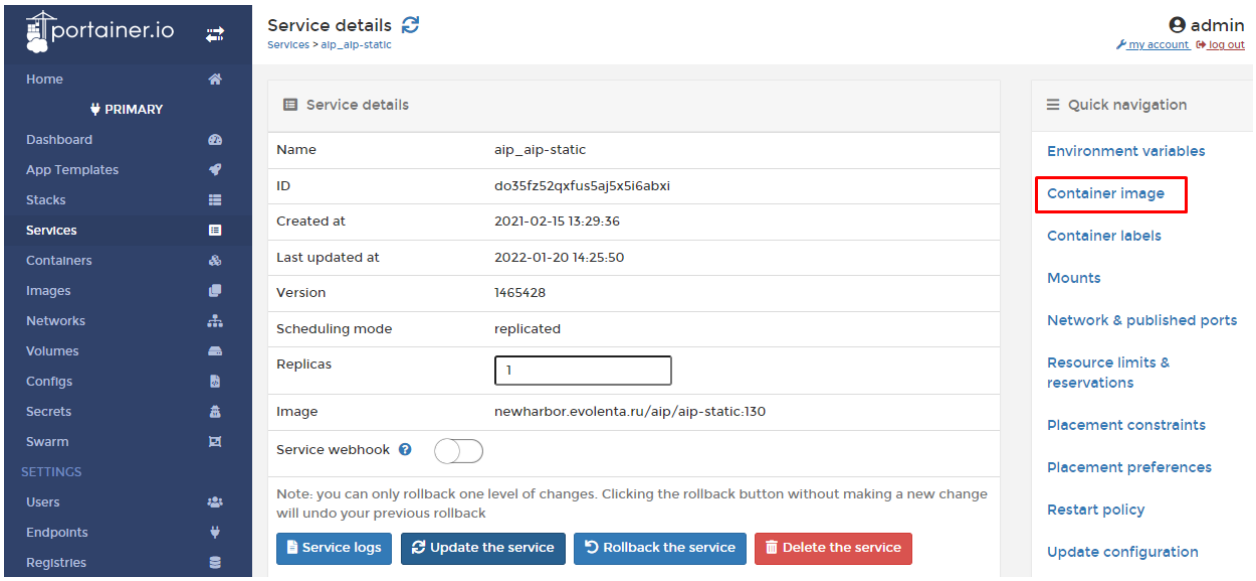

Рисунок 2 – Пункт меню «Container image» в программном обеспечении Portainer

4. В режиме «Simple mode» в поле «Image» указать новое имя образа – например, newharbor.evolenta.ru/sier/kpp:v46-mysql (Рисунок 3).

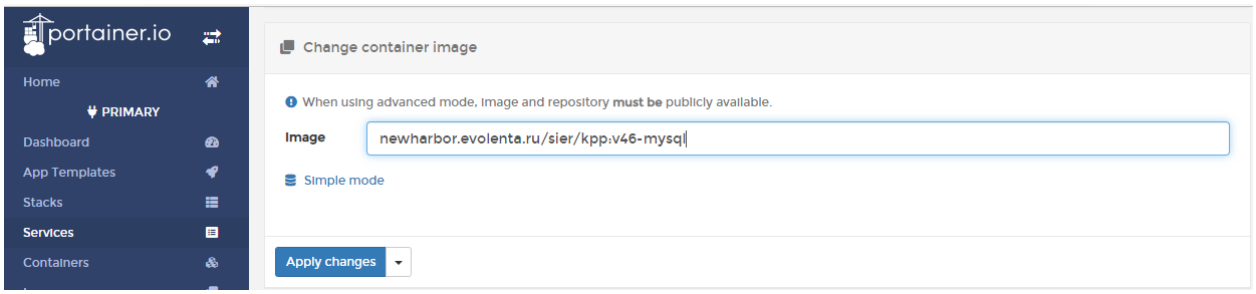

Рисунок 3 – Интерфейс пункта меню «Container image» в программном обеспечении Portainer

5. Нажать «Apply changes» в интерфейсе пункта меню «Container image» в программном обеспечении Portainer.

6. Перейти в пункт меню «Services» и проверить, что количество запущенных контейнеров соответствует требуемому (Рисунок 4).

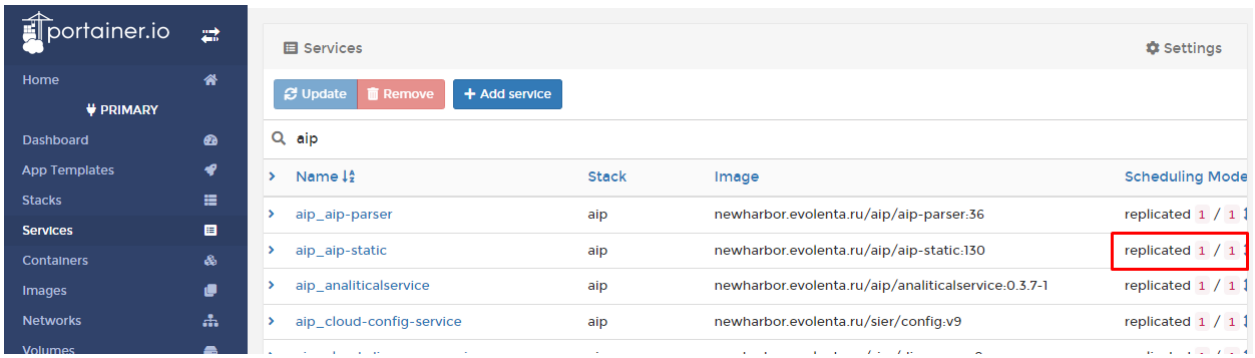

Рисунок 4 – Интерфейс пункта меню «Container image» в программном обеспечении Portainer

4.2. Выполнение резервного копирования

Поскольку все микросервисы Платформы являются сервисами без сохранения состояния (stateless), то в их резервном копировании нет необходимости. Однако микросервисы Платформы настроены на работу с внешними базами данных (БД), такими как MySQL, MongoDB, PosgtreSQL, которые нуждаются в резервировании.

Для резервного копирования БД используются штатные средства упомянутых СУБД, а именно команды: *mysqldump*, *mongodump*, *pg_dump*, соответственно. Ниже представлены примеры использования каждой команды:

1. Резервирование БД модуля авторизации и аутентификации КПП с последующей архивацией осуществляется следующей командой:

mysqldump -h ${MYSQL_HOST} -P ${MYSQL_PORT} -u ${MYSQL_USER} ${DATABASE_NAME} | gzip > archive_name.sql.gz,

где:

-h и -P – ip-адрес сервера и порт, на которых запущена СУБД MySQL,

-u – имя пользователя,

DATABASE_NAME – имя резервируемой БД.

2. Для резервирования БД системы аналитики Pentaho, БД платформы моделирования и создания бизнес-процессов Camunda, и БД геоданных, с

последующей архивацией используется следующая команда:

pg_dump –h ${PGSQL_HOST} –p ${PGSQL_PORT} -U ${PGSQL_USER} ${DATABASE_NAME} | gzip > archive_name.sql.gz,

где:

-h и -p – ip-адрес сервера и порт, на которых запущена СУБД PostgreSQL,

-U – имя пользователя,

DATABASE_NAME – имя резервируемой БД.

3. Резервное копирование БД системы обработки фотограмметрических данных (СОФД) с последующей архивацией выполняется следующей командой:

mongodump -h ${MONGO_HOST} --port ${MONGO_PORT} -u ${MONGO_USER} --authenticationDatabase ${AUTH_DATABASE_NAME} --db ${DATABASE_NAME} --gzip --archive= archive_name.gz,

где:

-h и --port – ip-адрес сервера и порт, на которых запущена СУБД MongoDB,

-u – имя пользователя,

--authenticationDatabase – БД, в которой авторизуется пользователь,

DATABASE_NAME – имя резервируемой БД.

Данные команды резервирования могут запускаться как вручную из командной строки, так и по расписанию с помощью команды crontab. Например, чтобы резервирование осуществлялось ежедневно в 01:00, нужно:

1) открыть планировщик crontab в режиме редактирования, путем ввода следующей команды:

sudo crontab –e

2) нажать клавишу I на клавиатуре и поместить туда следующую строку:

*00 01 * * * /путь/до/файла/mongo_backup.sh >/dev/null*,

где:

mongo_backup.sh – bash-скрипт с командой для резервирования БД Mongo.

3) нажать на клавиатуре

:wq,

чтобы сохранить изменения и выйти.

Проверить внесенные в планировщик изменения можно командой:

sudo crontab –l.

4.3. Осуществление проверки работоспособности Платформы

В качестве средств диагностики работы Платформы могут использоваться команды ОС Linux, а также сторонняя система мониторинга, построенная на системе сбора метрик «Prometheus» и системе визуализации «Grafana».

Проверка работоспособности Платформы с помощью командами ОС Linux осуществляется поочередным подключением к серверам, составляющим кластер Docker Swarm, по протоколу ssh, как показано в пункте 4.1.1, и выполнением команд:

1. *top*

На экран монитора будет выведено, кроме данных о доступной и используемой оперативной памяти (ОЗУ), файле подкачки и процессорах (для этого нужно нажать цифру 1 на клавиатуре), список процессов, наиболее сильно нагружающих Платформу. Пример результата использования команды ОС Linux *top* представлен на Рисунке 5.

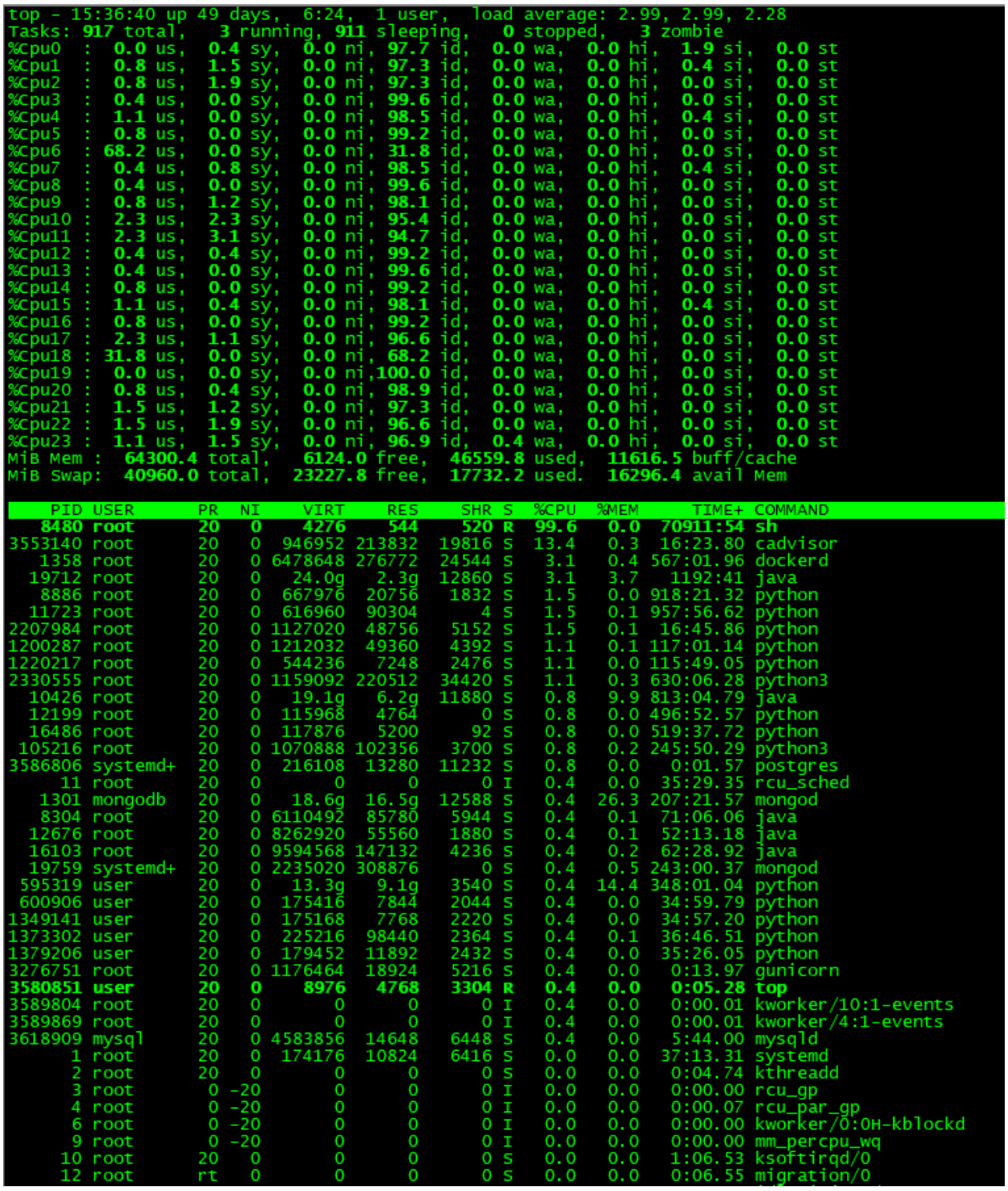

Рисунок 5 – Пример результата использования команды ОС Linux *top*

Следует обратить внимание на размер используемой оперативной памяти (Mib Mem / %Mem) и файла подкачки (Mib Swap / %Swap) – значения не должны быть выше 98% для обеспечения работоспособности Платформы.

Активное использование операционной системой (ОС) файла подкачки говорит о нехватки ОЗУ на вычислительных мощностях.

Также следует принять во внимание значение средней нагрузки (Load Average). Оптимальным считается значение, равное количеству ядер в ОС.

При значительном превышении указанных величин следует проанализировать лог-файлы ОС и приложений, как правило располагающиеся в директории */var/log*, и, при необходимости, выполнить оптимизацию используемого на сервере ПО либо добавить ОЗУ и увеличить мощность процессоров.

2. *sudo docker container stats*

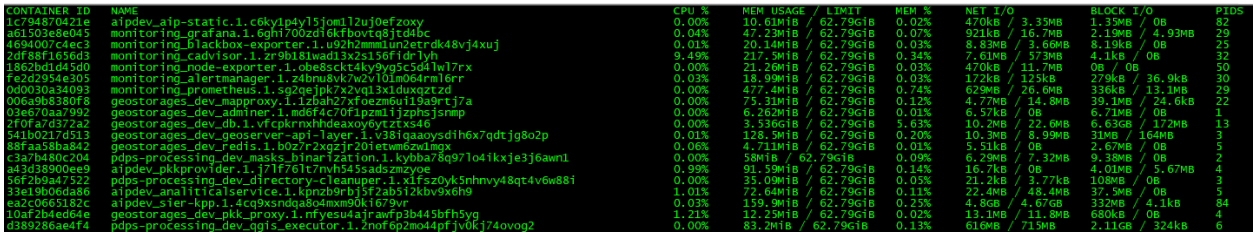

Рисунок 6 – Пример результата использования команды ОС Linux *sudo docker container stats*

Данная команда является полезной, если надо узнать сколько каждый из запущенных контейнеров Docker использует ОЗУ (Mem %) и процессорного времени (CPU %), а также сетевую активность (Net I/O) и активность на уровне блочного устройства (Block I/O).

Также для мониторинга работоспособности Платформы используется связка из сторонних систем «Prometheus» и «Grafana». Система сбора метрик «Prometheus» собирает разного рода метрики с серверов кластера Docker Swarm и запущенных на них контейнерах, на основе которых система визуализации «Grafana» строит графики.

Для доступа в графической оболочке системы «Grafana» нужно перейти по ссылке:

http://server-ip-address:3000,

где:

server-ip-address – ip-адрес или доменное имя сервера, на котором развернута система «Grafana».

Система «Grafana» отображает графики, касающиеся работы серверов. Пример визуализации графиков в системе «Grafana» представлен на рисунке 7.

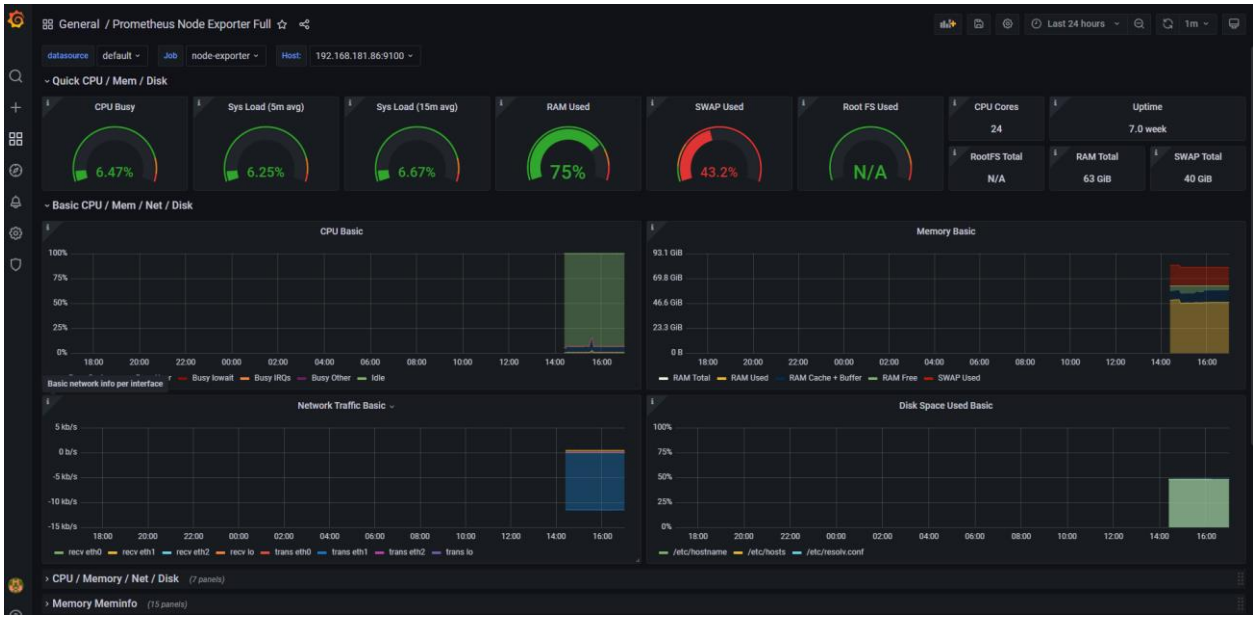

Рисунок 7 – Пример визуализации графиков в системе «Grafana»

Кроме того, система «Grafana» визуализирует детальную информацию, получаемую от контейнеров (представлено на рисунке 8).

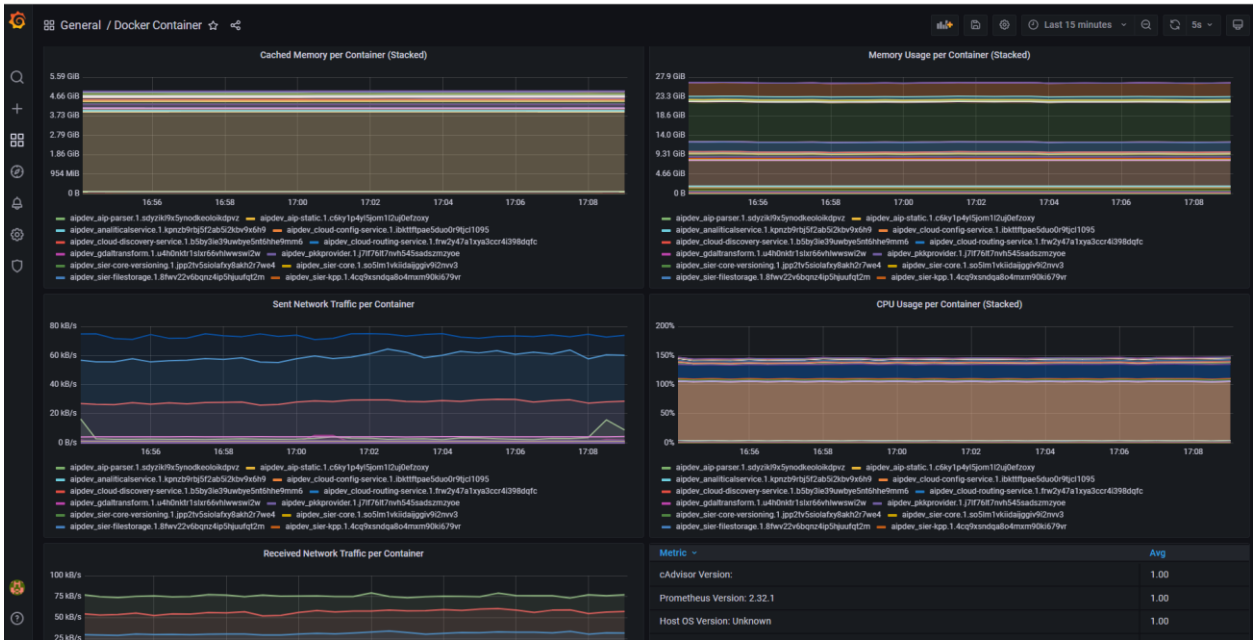

Рисунок 8 – Пример визуализации детальной информации, получаемой от контейнеров, в системе «Grafana»

Среди выводимых параметров (детальной информации, получаемой от контейнеров) в системе «Grafana» – использование ОЗУ, процессорного времени, средняя нагрузка, работа дисковой подсистемы, сетевой трафик и другие.

Основываясь на полученной информации, администратор в рамках технического обслуживания может увидеть проблемы в работе Платформы, а также спрогнозировать возможное появление проблем.

Указанная система мониторинга также способна отправлять оповещения на электронную почту или в мессенджеры при достижении контролируемыми параметрами пороговых значений.

4.4. Восстановление работоспособности Платформы при аварийных ситуациях

Восстановление работоспособности Платформы требует выявления причин возникновения аварийной ситуации. Для этого нужно проделать следующие действия:

1. Убедиться в работоспособности всех серверов в кластере Docker Swarm. Для этого необходимо подключиться к любому серверу кластера по протоколу ssh, как описано в пункте 4.1.1. настоящего документа, и выполнить команду:

sudo docker node ls

Пример результата использования команды ОС Linux *sudo docker node ls представлен на рисунке 9.*

Рисунок 9 – Пример результата использования команды ОС Linux *sudo docker node ls*

Следует внимательно ознакомиться с результатами использования команды ОС Linux *sudo docker node ls*, а именно обратить внимание на значения полей «Status» и «Availability» – должны быть значения «Ready» и «Active», соответственно. Если какое-то из значений отличается от указанных, то необходимо подключиться к соответствующему серверу по протоколу ssh и проверить состояние процесса docker путем ввода следующей команды:

sudo systemctl status docker

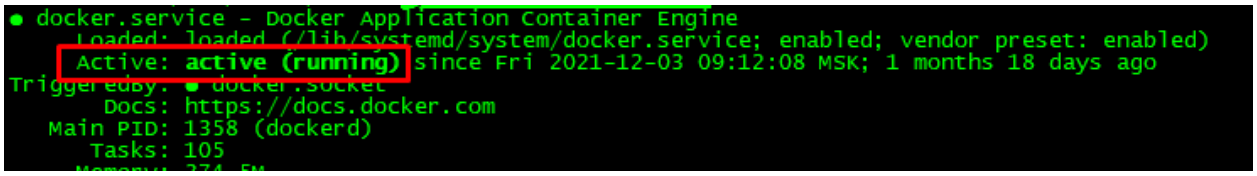

Рисунок 10 – Пример результата использования команды ОС Linux *sudo systemctl status docker*

Если в результатах использования команды ОС Linux *sudo systemctl status docker* статус отличается от «Active (Running)», то следует запустить службу *systemctl* путем ввода следующей команды:

sudo systemctl start docker

либо перезапустить данную службу в случае наличия ошибок путем ввода следующей команды:

sudo systemctl restart docker

2. Проверить состояние микросервисов. Для этого необходимо убедиться, что, в соответствии с пунктом 4.1.4 или 4.1.9, количество запущенных и запланированных к запуску реплик микросервиса одинаково (*replicated 1/1*). Если количество запущенных контейнеров равно нулю, то следует проанализировать лог-файлы микросервиса на наличие ошибок. Для этого необходимо подключиться к любому серверу кластера Docker Swarm по протоколу ssh, как описано в пункте 4.1.1, и выполнить команду:

sudo docker service logs service-name,

где

service-name – имя микросервиса, лог-файлы которого следует проанализировать.

3. Для просмотра лог-фалов из графической оболочки «Portainer» необходимо выполнить пункты 4.1.5 и 4.1.6. Затем нажать левой кнопки мыши на кнопку «>» для раскрытия списка контейнеров, и кнопку «Logs» в столбце «Actions». Интерфейс графической оболочки «Portainer», с указанием графических элементов для просмотра лог-фалов, представлен на рисунке 11.

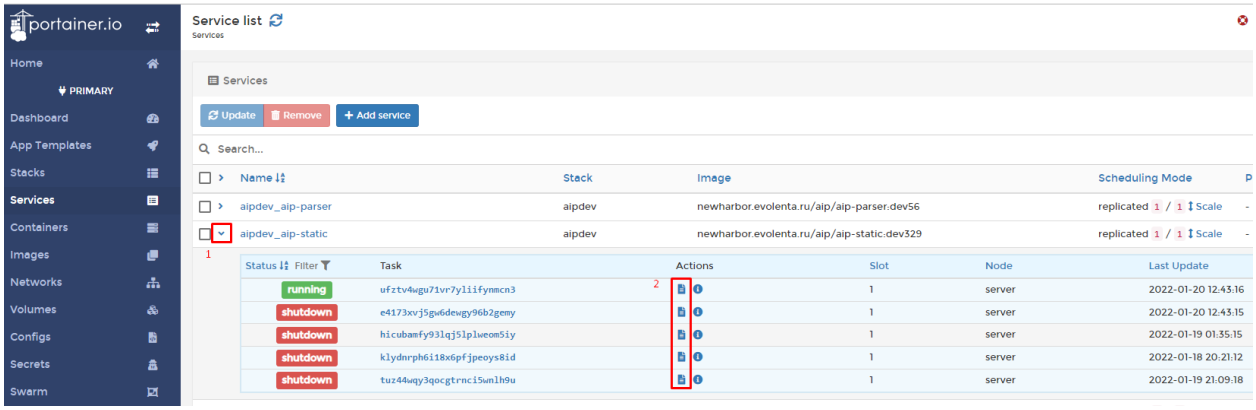

Рисунок 11 – Интерфейс графической оболочки «Portainer», с указанием графических элементов для просмотра лог-фалов

После совершения вышеописанных действия на экран монитора будет выведено содержимое лог-файла проблемного микросервиса, где по ключевому слову «Error» можно выявить причину неработоспособности. После устранения причины, контейнер будет запущен автоматически.

4. В случае, если контейнер микросервиса запущен, но анализ лог-файлов показал, что процесс внутри контейнера не функционирует должным образом, то следует осуществить перезапуск контейнера. Для этого из командной строки Linux нужно определить идентификатор (id) проблемного контейнера путем ввода следующей команды:

sudo docker container ls |grep aip-static

Пример результата использования команды ОС Linux *sudo docker container ls |grep aip-static*:

36a787413a73 *newharbor.evolenta.ru/aip/aip-static:dev329 "httpd-foreground" 23 hours ago Up 23 hours 80/tcp aipdev_aip-static.1.ufztv4wgu71vr7yliifynmcn3*

Затем использовать этот id для перезапуска контейнера путем ввода следующей команды:

sudo docker container kill 36a787413a73

и, в соответствии с пунктом 4.1.7, проверить, что количество запущенных контейнеров соответствует требуемому и что в лог-файлах отсутствуют ошибки.

Для выполнения перезапуска контейнера из графической оболочки «Portainer» необходимо выполнить пункты 4.1.5 и 4.1.6. Затем выбрать пункт меню «Containers», выделить проблемный контейнер, поставив галочку слева от его имени, и нажать на кнопку «Kill». Интерфейс графической оболочки «Portainer», с указанием графических элементов для выполнения перезапуска контейнера, представлен на рисунке 12.

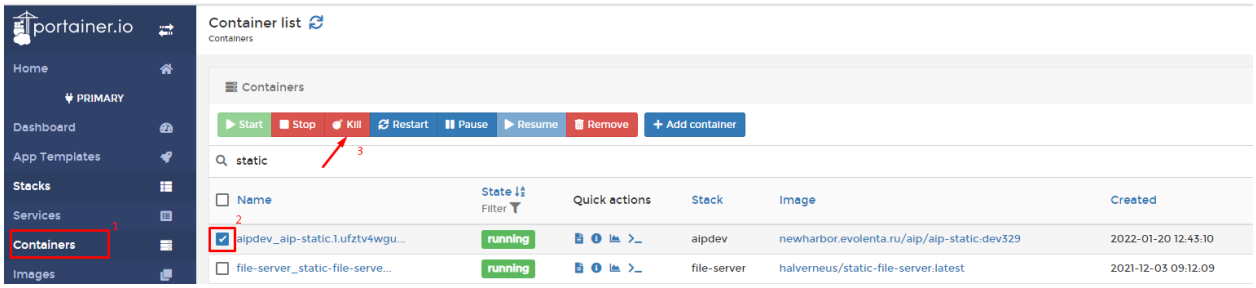

Рисунок 12 – Интерфейс графической оболочки «Portainer», с указанием графических элементов для выполнения перезапуска контейнера

После перезапуска контейнера из графической оболочки «Portainer» необходимо проверить сообщения в лог-файлах на наличие ошибок в соответствии с пунктом 4.4.3.

Лист регистрации изменений

Номера листов (страниц)					Всего листов (страниц) в докум	№ документа	Входящий № сопроводительного документа и дата	Подп.	Дата
Изм	измененных	замененных	новых	анулированных					