Digital Railways
Цифровые технологии
железных дорог

ООО «ЦТЖД», г.Санкт-Петербург
198095, ул. Маршала Говорова, 52А
info@rwdt.ru

Автоматизированная система «Система управления активами»

РУКОВОДСТВО ПО УСТАНОВКЕ

RU.08749795.02 94 03

2023

АННОТАЦИЯ

Настоящее руководство распространяется исключительно на программное обеспечение «Система управления активами» (далее – ПО, Система) и не заменяет учебную, справочную литературу, руководства от производителя Операционной Системы (далее – ОС) и прочие источники информации, освещающие работу с графическим пользовательским интерфейсом ОС.

В данном программном документе приведено руководство администратора по разворачиванию Системы и настройке её модулей.

В разделе «Назначение Системы» указано назначение Системы, а также отражены требования к подготовке администратора.

Раздел «Условия применения» содержит информацию о видах деятельности и функциях, автоматизация которых осуществляется Системой.

В разделе «Условия функционирования и обслуживания» указаны режимы функционирования Системы.

В разделе «Подготовка к работе» приведены состав и содержание дистрибутивного набора данных, порядок установки и запуска Системы.

СОДЕРЖАНИЕ

1. ОПРЕДЕЛЕНИЯ, ОБОЗНАЧЕНИЯ И СОКРАЩЕНИЯ

Термин/сокращение	Определение
БД	База данных
МТР	Материально-технический ресурс
Система, АС СУА	Автоматизированная система «Система управления активами»
СУБД	Система управления базами данных
УИИ	Уникальный идентификатор изделия
ЭВМ	Электронная вычислительная машина
API	Application Programming Interface - специальный протокол для взаимодействия компьютерных программ, который позволяет использовать функции одного приложения внутри другого

2. НАЗНАЧЕНИЕ СИСТЕМЫ

2.1. Функциональное назначение

Система предназначена для автоматизированного управления материальными активами предприятий.

2.2. Возможности Системы

Система позволяет учитывать, контролировать перемещение и эксплуатацию, а также отслеживать фактическое состояние объектов включая:

- автоматизированно формировать заявки на присвоение уникального идентификатора изделия материально-техническим ресурсам;
- автоматизированно заполнять УИИ с учетом различных вариантов согласования заявок на присвоение УИИ;
- устананавливать соответствие УИИ в единой базе данных с существующими классификаторами МТР (коды МТР, артикулы, заводские номера);
- формировать единую базу данных и хранения УИИ;
- осуществлять мониторинг прослеживаемости жизненного цикла в разрезе УИИ;
- обеспечивать передачу данных об изменении местоположения и состояния материальных активов;
- добавлять произвольное количество событий жизненного цикла для УИИ;
- обеспечивать данными для автоматизации процесса инвентаризации;
- осуществлять взаимодействие с программно-аппаратными комплексами записи и чтения маркировки;
- группировать данные для построения отчетности в разрезе используемых аналитик.

2.3. Требования к уровню подготовки администратора

Администратор должен обладать высоким уровнем квалификации и практическим опытом выполнения работ по установке, настройке и администрированию применяемых программных и технических средств.

3. УСЛОВИЯ ПРИМЕНЕНИЯ

Для полнофункциональной работы пользователя в Системе технические и программные средства автоматизированных рабочих мест должны обеспечивать установку и запуск Яндекс.Браузера 18.5 и выше.

4. УСЛОВИЯ ФУНКЦИОНИРОВАНИЯ И ОБСЛУЖИВАНИЯ

Система функционирует в следующих режимах:

- штатный режим;
- регламентный;
- режим восстановления работоспособности;
- режим ограниченной производительности (надежности).

Штатный режим является основным режимом функционирования, обеспечивающий выполнение задач Системы. В штатном режиме обеспечивается возможность функционирования Системы 24 часа в сутки, 7 дней в неделю, 365 дней в году.

Регламентный режим обеспечивает проведение регламентных работ, в котором один или все модули не выполняют свои функции. В регламентном режиме предусматривается извещение пользователей о недоступности Системы при попытке входа в процессе регламентных работ и обновления внешних, а также извещение внешних и внутренних систем о недоступности ресурса для передачи данных.

Режим восстановления работоспособности предусматривает полное или частичное отключение компонентов Системы, проведение диагностики компонентов Системы с использованием тестовых сигналов, восстановление работоспособности Системы, устранение неисправностей.

Режим ограниченной производительности (надежности) предусматривает отключение некоторых модулей Системы с целью диагностики и устранения неисправностей.

5. ПОДГОТОВКА К РАБОТЕ

5.1. Состав и содержание дистрибутивного набора данных Системы

Дистрибутивный набор данных представлен следующими сущностями:

– исходный модуль;

– загрузочный модуль.

5.2. Установка Модуля мониторинга

На сервере с предустановленной ОС Ubuntu 18.04 следует последовательно выполнить следующие операции по установке программного обеспечения.

Скрипты и темплейты настроек располагаются в репозитории по адресу:

http://10.21.XX.X/monitoring-template

Далее необходимо:

1. установить git

sudo apt install git -y

cd /opt

2. клонировать репозиторий

git clone git@10.21.XX.X /monitoring-template.git

3. переименовать и перейти в каталог

mv monitoring-template monitoring

cd monitoring

4. установить Docker

sudo apt install docker.io

sudo systemctl start docker

sudo systemctl enable docker

sudo usermod -aG docker ${USER}

Проверить правильность установки Docker можно проверить с помощью команды docker --version

5. установить систему управления базами данных PostgreSQL со встроенным модулем PostGIS, который позволяет работать с картографическими данными

$ sudo apt install postgis

Выполняя указания интерактивного помощника, следует установить пакет.

После установки необходимо настроить доступ к базам данным таким образом, чтобы к ней можно было подключиться из сети Docker и невозможно из внешней сети.

Для настройки следует выполнить команду ifconfig. Будет отображена информация по сетевым адресам. В перечне необходимо найти IP-адрес интерфейса docke0. Обычно это 172.17.0.1, но необходимо перепроверить адрес интерфейса.

Далее, в файл /etc/postgresql/10/main/pg_hba.conf добавляем следующую запись:

host all all 172.17.0.0/24 trust

В результате будет разрешено всем контейнерам системы подключаться к базе данных, которая находится на хосте вне контейнера.

6. создать пользователя и базу данных

Для того, чтобы создать пользователя в системе управления базами данных PostgreSQL необходимо авторизоваться под учетной записью пользователя postgres:

$ su - postgres

Далее создать нового пользователя, выполнив команду

createuser --interactive --pwprompt

После успешного создания пользователя в системе необходимо создать базу данных, где данный пользователь будет с необходимым набором привилегий:

$ createdb -0 <username> <dbname>

Где <username> – имя вновь созданного пользователя, а <dbname> желаемое название БД.

Затем следует включить расширения для поддержки UUID в базе данных:

psql -d <dbname>

CREATE EXTENSION IF NOT EXISTS «uuid-ossp»

\q

7. подготовить Docker stack. Склонировать репозиторий http://192.168.1.1/atlas-docker-stack на на хостовую машину. В зависимости от текущего окружения перейти в необходимую подпапку — production или staging.

В каждой папке находится .env файл, который управляет переменными окружения. Необходимо отредактировать .env файл и указать корректные данные для подключения к базе данных (пользователь, пароль, база данных).

В файле docker-stack.yml находится описание работы каждого из образов системы. В нем нам необходимо поменять бинд mainhost и указать тот ip адрес, что используется у docker0 интерфейса.

Далее необходимо перейти к запуску сервисов:

– инициализировать swarm,

docker swarm init

– подключиться к реестру образов на 10.21.XX.X:5000.

– выполнить команду

docker login

– инициализировать стек

— docker stack deploy --with-auth-registry -c docker-stack.yml atlas-<ENV_NAME>

Где ENV_NAME – название текущего окружения staging/dev/production.

После успешного выполнения команды необходимо проверить запуск всех сервисов. С помощью команды docker ps проверить статус всех запущенных сервисов.

8. запустить очереди сообщений. Для этого установить supervisord и настроить cron на выполнения команды в контейнере каждую секунду. Затем создать новый файл /etc/cron.d/atlas-scheduler со следующим содержимым:

* * * * * docker exec $(docker ps | grep -Eoh «\w*atlas-<ENV_NAME>_rest.[a-zA-Z0-9.]+") php artisan schedule:run --verbose >> /dev/null 2>&1

Где ENV_NAME – название текущего окружения staging/dev/production.

После создания нового файла в кроне, необходимо обновить журнал работы cron

$ sudo service cron reload

Далее необходимо установить supervisord, выполнив команду

$ sudo apt install supervisor

$ sudo service supervisor restart

Затем следует создать новый файл worker.sh со следующим содержимым:

#!/bin/bash

docker exec $(docker ps | grep -Eoh «\w*atlas-<ENV_NAME>_rest.[a-zA-Z0-9.]+") php artisan queue:work --verbose --sleep=3 --tries=3 —env=development

Где ENV_NAME – название текущего окружения staging/dev/production.

Далее необходимо создать новый файл /etc/supervisor/conf.d/atlas-supervisord.conf со следующим содержимым

[program:atlas-worker]

process_name=%(program_name)s_%(process_num)02d

command=<worker_path>

numprocs=4

stdout_logfile=/var/log/atlas-worker.log

stdout_logfile_maxbytes=0

stderr_logfile=/var/log/atlas-worker.error.log

stderr_logfile_maxbytes=0

stopsignal=KILL

user=admin

autorestart=true

startretries=3

Где <worker_path> – полный путь до вашего worker.sh.

Затем следует выполнить перезапуск супервизора командой

sudo service supervisor reread && sudo service supervisor restart.

После успешного выполнения всех этапов настройки система запустит очередь сообщений и все необходимые автоматизированные процессы.

9. настроить сервисы. В директории conf находятся файлы:

– prometheus.yml;

– alertmanager.yml;

– blackbox-exporter.yml;

– alert.rules.

Необходимо выполнить следующие действия:

1. настроить prometheus.yml: в соответствии с примером добавить сервера на которые установлен node-exporter. Запись может выглядеть как fqdn имя сервера, или IP адрес. Через : указывается порт на котором запущен экспортер.

Пример:

```
  - job_name: 'node_exporter'
    static_configs:
      - targets:
        - node-exporter:9100
        - 10.21.XX.X:9100
        - 10.10.1.1:9100
        - site.ru:443
```

В задаче blackbox добавить список сайтов для внешнего мониторинга.

Пример:

```
  - job_name: 'blackbox'
    metrics_path: /probe
    params:
      module: [http_2xx]  # Look for a HTTP 200 response.
    static_configs:
      - targets:
        - http://site1.ru:80
        - http://10.21.XX.X:8080
        - https://site3.ru:443
```

2. настроить alertmanager.yml: в соответствии с примером для отправки уведомлений о событиях указать параметры для smtp сервера, а именно: кому отправлять сообщения, с какого адреса, прописать адрес, порт и авторизацию на smtp сервере.

Пример:

```
receivers:
  - name: 'email_config'
    email_configs:
    - to: 'mail1@mail.ru, mail2@mail.ru'
      from: 'monitoring@mail.ru'
      smarthost: 'smtp.mail.ru:465'
      auth_username: 'username'
      auth_password: 'password'
      require_tls: false
```

3. файл blackbox-exporter.yml настроен на проверку веб сайтов, файл alert.rules – это триггеры для срабатывания аллертов.

Настроено 2 проверки для сайтов:

- доступен ли сайт;
- сертификат ssl истекает через неделю.

Настроены тригеры на ресурсы сервера:

- нод экспортер на сервере недоступен;
- средняя утилизация процессора последние 5 минут выше 80 %;
- iowait процессора за 5 минут выше 5 %;
- количество использованной памяти сервера выше 80 %;
- диск заполнен на 90 %.

Затем следует запустить сервис, выполнив скрипт:

```
sudo ./start.sh
```

На сервисы можно зайти по адресам

http://ip:3000 вебсервер grafana. По умолчанию логин пароль admin / admin

http://ip:9093 вебсервер alertmanager

http://ip:9100/metrics вебсервер метрик node-exporter

http://ip:9090 вебсервер prometheuse

http://ip:19999 вебсервер netdata

5.3. Установка node-exporter на сервера

На сервере необходимо установить node-exporter выполнив команды:

sudo apt install prometheus-node-exporter

systemctl restart prometheus-node-exporter

systemctl enable prometheus-node-exporter

Проверка осуществляется командой:

curl http://127.0.0.1:9100/metrics

5.4. Установка Модуля логирования

Скрипты и темплейты настроек лежат в репозитории по адресу:

http://10.21.XX.X/elk-template

Далее необходимо:

1. установить git

sudo apt install git -y

cd /opt

2. клонировать репозиторий

git clone git@192.168.1.1/elk-template.git

3. переименовать и перейти в каталог

mv elk-template elk

cd elk

4. запустить скрипт установки docker и docker-compose

sudo ./install.sh

Затем следует настроить сервисы. В директории conf находятся файлы:

- elasticsearch.yml;
- kibana.yml;
- logstash.conf;
- logstash.yml.

В файле config/logstash.conf необходимо изменить IP адрес брокера кафки на fqdn запись или IP адрес сервера кафки.

Пример

```
        kafka {
                codec => json
                topic_id => "logstash"
                bootstrap_servers => "192.168.1.16667"
                batch_size => 1000
        }
```

Затем следует запустить сервис, выполнив скрипт:

sudo ./start.sh

Веб серверы доступны по адресам:

http://ip:5601 сервис визуалиазиции Kibana

http://ip:9200 API Elasticsearch

5.5. Установка filebeat на сервера

Для установки filebeat на сервере необходимо выполнить команды:

curl -L -O https://artifacts.elastic.co/downloads/beats/filebeat/filebeat-7.4.0-amd64.deb

sudo dpkg -i filebeat-7.4.0-amd64.deb

Для настройки filebeat необходимо изменить файл /etc/filebeat/filebeat.yml

```
filebeat.inputs:

- type: log
  enabled: true
  paths:
    - /var/log/syslog
  fields:
    type: syslog
  fields_under_root: true
  scan_frequency: 5s
- type: log
  enabled: true
```

```
  paths:
    - /var/log/auth.log
  fields:
    type: auth
  fields_under_root: true
  scan_frequency: 5s

output.logstash:
  hosts: ["10.21.XX.X:5044"]
xpack:
  enabled: false
name: elk
logging.level: error
logging.to_files: true
logging.files:
  path: /var/log/filebeat
  name: filebeat.log
  rotateeverybytes: 10485760 # = 10MB
  keepfiles: 7
  rotateonstartup: true
name: server_name
```

В раздел filebeat.inputs
Надо добавить источники логов

```
- type: log
  enabled: true
  paths:
    - /var/log/syslog
  fields:
```

```
  type: syslog
fields_under_root: true
scan_frequency: 5s
```

Необходимо изменить поле type: на имя сервиса, который генерирует лог. Для этого выполнить следующие действия:

– в поле paths: прописать полный путь до файла. Путь может состоять из нескольких записей и включать маску*;

– в разделе output.logstash изменить IP адрес, на адрес сервера Logstash

– hosts: ["10.21.XX.X:5044"]

– поле name надо заменить на fqdn имя сервера.

name: server.name.ru

После настройки надо перезапустить сервис filebeat, выполнив команды:

sudo update-rc.d filebeat defaults 95 10

sudo service filebeat restart

5.6. Установка стека ZooKeeper, Kafka, Ni-Fi

Для установки необходимо установить на компьютер администратора ansible и git выполнив команду:

apt install ansible git -y

Затем следует клонировать репозиторий с проектом:

git clone git@10.21.XX.X:a.melnikov/cifra_infra.git

cd cifra_infra

Для запуска необходимо минимум 9 серверов с ubuntu 18:

3 сервера для кластера docker-swarm и zookeeper;

3 сервера kafka;

3 сервера nifi.

Далее необходимо внести изменения в файл hosts.yml:

1. изменить параметры доступа к серверам:

```
all:
  vars:
    ansible_user: user
    ansible_port: 22
    ansible_ssh_pass: 1234
    ansible_become_pass: 1234
    ansible_become: true
```

Аналогичным образом поменять имя пользователя и пароли, а также порт SSH если он отличается от 22 для доступа к новым 9 серверам.

2. поменять IP адреса серверов:

```
  hosts:
    ansible-swarm-01:
      ansible_host: 10.21.XX.X
      ansible_port: 22001

    ansible-swarm-02:
      ansible_host: 10.21.XX.X
      ansible_port: 22002

    ansible-swarm-03:
      ansible_host: 10.21.XX.X
      ansible_port: 22003

    ansible-kafka-01:
      ansible_host: 10.21.XX.X
      ansible_port: 22004
      kafka_broker_id: 0

    ansible-kafka-02:
      ansible_host: 10.21.XX.X
```

```
    ansible_port: 22005
    kafka_broker_id: 1

  ansible-kafka-03:
    ansible_host: 10.21.XX.X
    ansible_port: 22006
    kafka_broker_id: 2

  ansible-nifi-01:
    ansible_host: 10.21.XX.X
    ansible_port: 22007

  ansible-nifi-02:
    ansible_host: 10.21.XX.X
    ansible_port: 22008

  ansible-nifi-03:
    ansible_host: 10.21.XX.X
    ansible_port: 22009
```

В записи указываются следующие параметры:

ansible-swarm-01 – имя сервера, должно быть полное fqnd имя;

ansible_host: 10.21.XX.X – IP адрес сервера;

ansible_port: 22001 – порт сервера.

3. изменить группы серверов:

```
docker_ce:
  hosts:
    ansible-swarm-01:
    ansible-swarm-02:
    ansible-swarm-03:
```

```
docker_swarm:
  children:
    docker_ce:

swarm_zookeeper:
  children:
    docker_swarm:

apache_kafka:
  hosts:
    ansible-kafka-01:
    ansible-kafka-02:
    ansible-kafka-03:

apache_nifi:
  hosts:
    ansible-nifi-01:
    ansible-nifi-02:
    ansible-nifi-03:
```

В записи указываются имена групп:
docker_ce;
docker_swarm;
swarm_zookeeper;
apache_kafka;
apache_nifi.
В группу может входить отдельный хост или дочерняя группа.
Пример записи для отдельных хостов:
docker_ce:

```
  hosts:
    ansible-swarm-01:
    ansible-swarm-02:
    ansible-swarm-03:
```

Пример записи для дочерней группы:

```
docker_swarm:
  children:
    docker_ce:
```

После внесения изменений необходимо проверить синтаксис файла:

– yamllint hosts.yml – проверит файл на соответствие стандартам YML;

– ansible-lint site.yml -v проверяет весь проект на соответствие стандартам ansible. Поле прохода тестов, если нет критичных ошибок, можно стартовать установку сервисов;

– ansible -i hosts.yml all -m ping – проверка доступов к серверам.

Если все тесты прошли успешно – можно запускать установку:

ansible-playbook site.yml -i hosts.yml -D

Первая установка должна занимать около 40 минут из-за скачивания дистрибутивов Kafka, Ni-Fi из Интернета.

Следующие проходы для изменений настроек занимают около 1-2 минут.

При выполнении ansible выводит информацию с именем задачи, сервера, а также показывает сделанные изменения.

По окончанию установки ansible выведет статистику по выполненным задачам.

Пример

```
PLAY RECAP *********************************************************************
ansible-kafka-01 : ok=11 changed=0 unreachable=0 failed=0 skipped=3 rescued=0 ignored=0
```

ansible-kafka-02 : ok=11 changed=0 unreachable=0 failed=0 skipped=3 rescued=0 ignored=0

ansible-kafka-03 : ok=11 changed=0 unreachable=0 failed=0 skipped=3 rescued=0 ignored=0

ansible-nifi-01 : ok=12 changed=0 unreachable=0 failed=0 skipped=3 rescued=0 ignored=0

ansible-nifi-02 : ok=12 changed=0 unreachable=0 failed=0 skipped=3 rescued=0 ignored=0

ansible-nifi-03 : ok=12 changed=0 unreachable=0 failed=0 skipped=3 rescued=0 ignored=0

ansible-swarm-01 : ok=21 changed=1 unreachable=0 failed=0 skipped=1 rescued=0 ignored=0

ansible-swarm-02 : ok=18 changed=0 unreachable=0 failed=0 skipped=1 rescued=0 ignored=0

ansible-swarm-03 : ok=18 changed=0 unreachable=0 failed=0 skipped=1 rescued=0 ignored=0

В поле failed должно быть 0 у всех серверов. Если есть ошибки необходимо изучить лог и исправить ошибки на сервере.

После прохода ansible будут настроены все сервера. Zookeeper, kafka, nifi будут настроены на доступ на своих стандартных портах.

Для настройки мониторинга следует добавить 2 группы в hosts.yml: node_exporter_sytemd, filebeat_systemd со списком хостов, на которых надо установить node-exporter и filebeat.

Затем необходимо изменить адрес logstash сервера и пути до лог файлов и имена сервисов.

Пример

```
all:
  vars:
    elk_elastic_address: 10.21.XX.X:5044
    filebeat:
```

```
    - name: syslog
      files:
        - /var/log/syslog
    - name: auth
      files:
        - /var/log/auth.log
```

Можно эти параметры поменять для всех серверов командой:

```
all:
  vars:
```

Или сделать изменение для 1 сервера:

```
  ansible-nifi-01:
    ansible_host: 10.21.XX.X
    elk_elastic_address: 10.21.XX.X:5044
    filebeat:
      - name: syslog
        files:
          - /var/log/syslog
      - name: auth
        files:
          - /var/log/auth.log
      - name: app
        files:
          - /var/log/app/access.log
```

После этих изменений еще раз н запустить команду:

```
ansible-playbook site.yml -i hosts.yml -D
```