

Решаем истинные задачи

Руководство администратора Системы

Содержание

[1 Перечень программного обеспечения для установки WFM 2](#_Toc95490752)

[2 Инструменты администрирования Системы 4](#_Toc95490753)

[3 Схема развертывания и настройки системы сотрудником зказчика 5](#_Toc95490754)

[3.1 Настройка сервера тестового контура 5](#_Toc95490755)

[3.2 Настройка базы данных redis 5](#_Toc95490756)

[3.3 Настройка сервера базы данных WFM в тестовом контуре 6](#_Toc95490757)

[3.4 Настройка сервера приложений в тестовом контуре 6](#_Toc95490758)

[3.5 Настройка универсального интеграционного адаптера 11](#_Toc95490759)

[3.6 Перенос данных с тестового контура на продуктивный 12](#_Toc95490760)

[3.7 Создание произвольных отчетных форм через конструктора отчетов 14](#_Toc95490761)

[4 Настройка типов авторизации 16](#_Toc95490762)

[4.1 Журналирование событий 16](#_Toc95490763)

[4.2 Настройка отправки уведомлений 22](#_Toc95490764)

[4.3 Порядок остановки и запуска Системы 22](#_Toc95490765)

[5 Настройка резервного копирования 23](#_Toc95490766)

[6 Поиск и устранение неисправностей 24](#_Toc95490767)

[7 проверка функционирования системы 26](#_Toc95490768)

[Приложение 1 ОПИСАНИЕ КОНФИГУРАЦИОННОГО ФАЙЛА СИСТЕМЫ 28](#_Toc95490769)

1. Перечень программного обеспечения для установки WFM

Для функционирования WFM системы и решения функциональных задач подходят вычислительные мощности как выделенного, так и виртуализируемого серверного оборудования. Для более качественной настройки программного и аппаратного обеспечения, а также для отладки всех интеграционных процессов и внедрению новых релизов Системы рекомендуется развернуть аппаратную архитектуру, способную обеспечить бесперебойную работу программного продукта в тестовой и продуктивной среде. Ниже будут приведены требования к конфигурации серверного оборудования, исходя из следующих данных:

* Общее количество операторов/количество одновременно работающих – 4933/3711;
* Общее количество менеджеров/количество одновременно работающих – 265/169

Таблица 1. Требования к конфигурации оборудования

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **#** | **Назначение** | **Количество** | **Процессор** | **ОЗУ** | **Дисковое пространство** | **Используемое ПО** |
| 1 | Сервер приложений продуктивный контур | 1 | 8 cores 2.2 GHz | 64 Gb | 256Gb | РЕД ОС, Nginx, Apache-Tomcat 9, Redis 6 |
| 2 | Сервер базы данных продуктивный контур | 1 | 8 cores 2.2 GHz | 32 Gb | 2 x 1024 Gb | РЕД ОС, postgresql 12 |
| 3 | Сервер приложений тестовый контур | 1 | 4 cores 2.2 GHz | 16 Gb | 60 Gb | РЕД ОС, Nginx, Apache-Tomcat 9, Redis 6 |
| 4 | Сервер баз данных тестовый контур | 1 | 4 cores 2.2 GHz | 16 Gb | 256 Gb | РЕД ОС, postgresql 12 |

Для обеспечения комфорта работы пользователей системы рекомендуется обеспечить скорость передачи данных по каналам связи от 100 Мбит.

В качестве системного программного обеспечения используется дистрибутив Linux РЕД ОС.

Для хранения и обработки данных в Системе WFM используется объектно-реляционная система управления базами данных Postgresql.

Для обеспечения функционирования всех компонентов системы WFM используются следующие операционные системы, а также средства расширяющие возможности операционных систем:

Таблица 2. Описание программного обеспечения

| # | Наименование ПО | Описание ПО |
| --- | --- | --- |
| 1 | РЕД ОС | РЕД ОС является дистрибутивом GNU/Linux, основанном на открытом исходном коде и собственных разработок российских производителей. |
| 2 | PostgreSQL | объектно-реляционная система управления базами данных. Она поддерживает большую часть стандарта SQL и предлагает множество современных функций: сложные запросы, внешние ключи, триггеры, изменяемые представления, транзакционная целостность, многоверсионность |
| 3 | Apache Tomcat | контейнер сервлетов с открытым исходным кодом, разрабатываемый̆ Apache Software Foundation. Реализует спецификацию сервлетов, спецификацию JavaServer Pages (JSP) и JavaServer Faces (JSF). Apache Tomcat позволяет запускать веб-приложения, содержит ряд программ для самоконфигурирования. Данное программное средство используется в качестве самостоятельного веб-сервера, в качестве сервера контента в сочетании с веб- сервером Apache HTTP Server. |
| 4 | JDK | комплект разработчика приложений на языке Java, включающий в себя компилятор Java, стандартные библиотеки классов Java, примеры, документацию, различные утилиты и исполнительную систему Java. |
| 5 | Nginx | веб-сервер и почтовый̆ прокси-сервер, работающий̆ на Unix-подобных операционных системах, в частности, на Linux |
| 6 | Redis | резидентная система управления базами данных класса NoSQL с открытым исходным кодом, работающая со структурами данных типа «ключ — значение». |

1. Инструменты администрирования Системы

Для полноценного администрирования системы пользователю, обладающему соответствующими правами, потребуется осуществлять подключение к серверу приложений по протоколу SSH и к системе управления базы данных PostgreSQL.

При использовании, в качестве рабочего места администратора операционной системы семейства Windows, потребуется установить свободно распространяемый клиент для сетевых протоколов SSH, telnet и Rlogin – putty. Скачать установочный файл можно по ссылке [https://putty.org.ru](https://putty.org.ru/). На этом же сайте можно подробней ознакомиться с возможностями данной программы.

Для осуществления подключения к системе управления базами данных PostgreSQL, удобно использовать инструмент pgAdmin, являющийся открытой платформой администрирования и разработки для СУБД PostgreSQL. Ссылка на официальный сайт решения [https://www.pgadmin.org](https://www.pgadmin.org/).

При использовании Unix-подобной операционной системы в качестве рабочего места администратора, возможно обойтись стандартными средствами, а именно встроенной консолью (терминалом). В большинстве Unix-подобных дистрибутивов все необходимые инструменты предустановлены.

Предполагается, что пользователь, выбравший в качестве рабочего места Unix-подобную операционную систему, владеет достаточной компетенцией для самостоятельной настройки подключения к серверу приложений и системы управления базами данных программного обеспечения WFM.

1. Схема развертывания и настройки системы сотрудником зказчика
   1. Настройка сервера тестового контура

Все описанные в настоящем документе действия, предполагают наличия у пользователя прав администратора. Тестовый контур представляет из себя виртуальный или физический сервер с установленной операционной системой. Предполагается, что сервер приложений и сервер базы данных, будут развернуты на одном сервере, при использовании продуктивного контура, предполагается разнести компоненты Системы на разные сервера в качестве горизонтального масштабирования.

На сервер тестового контура устанавливается система управления базы данных, сервер приложений и веб-сервер nginx.

Установка **WFM** производится на сервер под управлением операционной системой РЕД ОС. Минимально необходимый набор установленных компонентов и пакетов:

* Netwoking;
* SSH(22 порт);
* htop, iotop, iftop;
* Наличие файлового менеджера Midnight Commander;
* UnixODBC;
* Tomcat 8;
* JDK8;
* PostgreSQL;
* Redis

После установки ОС необходимо настроить корпоративные межсетевые экраны для обеспечения доступа пользователей и администраторов. Для доступа используются следующие порты:

* TCP/22 (ssh, доступ для системных администраторов);
* TCP/5432 (PostgreSQL по умолчанию, доступ для сервера приложений, DBA);
* TCP/8080 (web-сервер по умолчанию, сервер приложений **WFM**);
* TCP/8443 (web-сервер с SSL, по умолчанию).
  1. Настройка базы данных redis

Для настройки Redis требуется отредактировать конфигурационный файл, расположенный, по умолчанию, в каталоге /etc/redis.conf.

Что бы осуществлять удаленное подключение к БД Redis, требуется в конфигурационном файле найти запись bind 127.0.0.1 и добавить через пробел IP адрес хоста, с которого будет осуществляться удаленное подключение.

*Пример*

bind 127.0.0.1 192.168.1.10

Далее требуется установить пароль для подключения к БД Redis, для чего в конфигурационном файле, в разделе SECURITY необходимо найти строчку # requirepass foobared и убрать знак комментария "#", а после слова requirepass указать выбранный пароль через пробел.

*Пример*

requirepass nash\_password

Следующим шагом требуется сохранить изменения и выполнить перезапуск Redis.

Далее необходимо внести данные для подключения к redis в основной конфигурационный файл WFM - wfmserver.properties, а именно добавить в конце файла три настройки:

* spring.redis.host = localhost #Адрес сервера где запущен Redis
* spring.redis.port = 6379 # Порт, который слушает Redis
* spring.redis.password = # Пароль, указанный в конфигурационном файле Redis по адресу /etc/redis.conf
  1. Настройка сервера базы данных WFM в тестовом контуре

После того как на сервер базы данных установлена СУБД PostgreSQL, требуется создать пустую базу данных для работы **WFM**. Для этого пользователем postgres из консоли psql требуется создать пользователя **tomcat**:

* CREATE ROLE tomcat VALID UNTIL 'infinity';
* ALTER ROLE "tomcat" with login;
* Пользователю требуется задать пароль (вместо some\_password):
* ALTER USER tomcat WITH PASSWORD 'some\_password';
* Далее создается пустая база данных, принадлежащая пользователю tomcat:
* CREATE DATABASE wfm WITH ENCODING='UTF8' OWNER=tomcat CONNECTION LIMIT=-1;

Следующим шагом требуется осуществить подключение к только что созданной базе данных и выполнить установку расширения "uuid-ossp", которое предоставляет функции для генерирования универсальных уникальных идентификаторов (UUID) по одному из нескольких стандартных алгоритмов. В нём также есть функции, выдающие специальные UUID-константы. для этого необходимо выполнить следующие команды в консоли:

\c wfm

create extension "uuid-ossp"

Далее следует настроить права подключения пользователей к базам данных, текущего кластера. Данные настройки производятся в **/var/lib/pgsql/10.0/data/pg\_hba.conf**

Рекомендуется выставить метод md5, подразумевающий ввод пароля, для локального подключения и для подключения с удаленного хоста.

local all all md5

host all all 0.0.0.0/0 md5

Для возможности подключения с удаленного хоста также следует в файл postgresql.conf добавить следующие строки:

* listen\_addresses = '\*'
* port = 5432
  1. Настройка сервера приложений в тестовом контуре

В отличие от Windows, файловая система Linux не находится в одной папке а, как правило, распределена по корневой файловой системе. Это распределение поддается определенным правилам. Назначение каждой директории регламентирует «Стандарт иерархии файловой системы» FHS (Filesystem Hierarchy Standard). Ниже опишем основные директории согласно стандарту FHS.

Перечень системных каталогов:

/ — root каталог. Содержит в себе всю иерархию системы;

/bin — содержит двоичные исполняемые файлы. Основные общие команды, хранящиеся отдельно от других программ в системе (прим.: pwd, ls, cat, ps);

/boot — содержит файлы, используемые для загрузки системы (образ initrd, ядро vmlinuz);

/dev — в данной директории располагаются файлы устройств (драйверов). С помощью этих файлов можно взаимодействовать с устройствами. К примеру, если это жесткий диск, можно подключить его к файловой системе. В файл принтера же можно написать напрямую и отправить задание на печать;

/etc — в этой директории находятся файлы конфигураций программ. Эти файлы позволяют настраивать системы, сервисы, скрипты системных демонов;

/home — каталог, аналогичный каталогу Users в Windows. Содержит домашние каталоги учетных записей пользователей (кроме root). При создании нового пользователя здесь создается одноименный каталог с аналогичным именем и хранит личные файлы этого пользователя;

/lib — содержит системные библиотеки, с которыми работают программы и модули ядра;

/lost+found — содержит файлы, восстановленные после сбоя работы системы. Система проведет проверку после сбоя и найденные файлы можно будет посмотреть в данном каталоге;

/media — точка монтирования внешних носителей;

/mnt — точка временного монтирования. Файловые системы подключаемых устройств обычно монтируются в этот каталог для временного использования;

/opt — тут расположены дополнительные (необязательные) приложения. Такие программы обычно не подчиняются принятой иерархии и хранят свои файлы в одном подкаталоге (бинарные, библиотеки, конфигурации), именно в эту директорию будет выполнена установка сервера приложения WFM системы;

/proc — содержит файлы, хранящие информацию о запущенных процессах и о состоянии ядра ОС;

/root — директория, которая содержит файлы и личные настройки суперпользователя;

/run — содержит файлы состояния приложений. Например, PID-файлы или UNIX-сокеты;

/sbin — аналогично /bin содержит бинарные файлы. Утилиты нужны для настройки и администрирования системы суперпользователем;

/srv — содержит файлы сервисов, предоставляемых сервером (прим. FTP или Apache HTTP);

/sys — содержит данные непосредственно о системе. Здесь возможно узнать информацию о ядре, драйверах и устройствах;

/tmp — содержит временные файлы. Данные файлы доступны всем пользователям на чтение и запись. Стоит отметить, что данный каталог очищается при перезагрузке;

/usr — содержит пользовательские приложения и утилиты второго уровня, используемые пользователями, а не системой. Содержимое доступно только для чтения (кроме root). Каталог имеет вторичную иерархию и похож на корневой;

/var — содержит переменные файлы. Имеет подкаталоги, отвечающие за отдельные переменные. Например, логи будут храниться в /var/log, кэш в /var/cache, очереди заданий в /var/spool/ и так далее.

Приступим к настройке самого сервер приложений. После настройки сервера базы данных, требуется установить веб-сервер Apache-Tomcat, на момент написания текущей инструкции последняя версия Tomcat 9.0.30. Далее приведены команды, которые следует запускать через консоль с правами администратора (root).

sudo yum install wget

Данная команда установит консольную программу «wget» для загрузки файлов по сети. Обычно программа «wget» является предустановленной на большинстве дистрибутивов Linux, но, если по каким-то причинам указанная программа отсутствует, необходим выполнить ее установку.

Далее sudo wget http://apache-mirror.rbc.ru/pub/apache/tomcat/tomcat-9/v9.0.30/bin/apache-tomcat-9.0.30.tar.gz

С помощью установленной ранее программой «wget», выполняется скачивание архива веб сервера Apache-Tomcat.

При помощи предустановленного архиватора tar, выполняется разархивирование скаченного архива.

sudo tar -xvzf apache-tomcat-9.0.30.tar.gz

При помощи команды «mv» осуществляется перемещение папки apache-tomcat-9.0.30 из домашнего каталога пользователя в директорию /opt с одновременным переименованием каталога в tomcat-wfm.

sudo mv apache-tomcat-9.0.30 /opt/tomcat-wfm

Структура системных каталогов tomcat-wfm является следующей:

* каталог bin – содержит управляющие скрипты, в частности, для запуска и остановки приложения;
* каталог conf – содержит конфигурационные файлы;
* каталог lib – основные программные библиотеки, используемые всеми расширениями Tomcat и веб-приложениями;
* каталог logs – каталог предназначен для хранения лог-файлов;
* каталог temp – содержит временные файлы;
* каталог webapps – сожержит файлы приложения;
* каталог work – рабочий каталог Tomcat, который используется, в первую очередь, при преобразовании JSP-страниц в сервлеты.

После настройки сервера базы данных, требуется установить веб-сервер Apache-Tomcat, на момент написания текущей инструкции последняя версия Tomcat 9.0.30. Далее приведены команды, которые следует запускать через консоль с правами администратора (root).

sudo yum install wget

Данная команда установит консольную программу «wget» для загрузки файлов по сети. Обычно программа «wget» является предустановленной на большинстве дистрибутивов Linux, но, если по каким-то причинам указанная программа отсутствует, необходим выполнить ее установку.

Далее sudo wget http://apache-mirror.rbc.ru/pub/apache/tomcat/tomcat-9/v9.0.30/bin/apache-tomcat-9.0.30.tar.gz

С помощью установленной ранее программой «wget», выполняется скачивание архива веб сервера Apache-Tomcat.

При помощи предустановленного архиватора tar, выполняется разархивирование скаченного архива.

sudo tar -xvzf apache-tomcat-9.0.30.tar.gz

При помощи команды «mv» осуществляется перемещение папки apache-tomcat-9.0.30 из домашнего каталога пользователя в директорию **/opt** с одновременным переименованием каталога в **tomcat-wfm**.

sudo mv apache-tomcat-9.0.30 /opt/tomcat-wfm

Структура системных каталогов tomcat-wfm является следующей:

* каталог bin – содержит управляющие скрипты, в частности, для запуска и остановки приложения;
* каталог conf – содержит конфигурационные файлы;
* каталог lib – основные программные библиотеки, используемые всеми расширениями Tomcat и веб-приложениями;
* каталог logs – каталог предназначен для хранения лог-файлов;
* каталог temp – содержит временные файлы;
* каталог webapps – сожержит файлы приложения;
* каталог work – рабочий каталог Tomcat, который используется, в первую очередь, при преобразовании JSP-страниц в сервлеты.

Следующим шагом требуется скачать и разархивировать программное обеспечение WFM для того, чтобы получить ПО, обратитесь к Исполнителю. В случае выхода новой версии программного обеспечения компания разработчик направляет Заказчику электронное письмо, содержащее соответствующую информацию.

После получения и разархивирования полученного образца специального программного обеспечения, у Вас появится каталог release27 (цифровое обозначение может меняться в зависимости от версии специального ПО) следующего содержания:

* каталог ROOT;
* архив ROOT.war;
* архив wfmserver.war;
* файл setenv.sh;
* файл wfmserver.propertes.

Далее необходимо прописать настройки подключения к ранее созданной базе данных в основном конфигурационном фале системы **wfmserver.properties.**

dataSource.url=jdbc:postgresql://ip\_addres:5432/wfm

dataSource.username=tomcat

dataSource.password=some\_password

Подробное описание настроек конфигурационного файла приведено в приложении 1.

После сохранения изменений необходимо выполнить перенос изменённого файла настройки подключения в директорию с веб-сервером следующей командой:

sudo mv wfmserver.properties /opt/tomcat-wfm/conf/ wfmserver.properties

Следующим шагом необходимо в файле **setenv.sh**, в переменной **CATALINA\_IPTS** прописать корректный путь к файлу настроек, который на прошлом шаге был перенесен в каталог веб-сервера.

Так же необходимо в файле **setenv.sh** в переменной J**AVA\_HOME** прописать корректный путь к домашней директории Java-машины.

Для того, чтобы узнать полный путь к домашней директории Java-машины выполните в консоли следующую команду:

which java

После сохранения выполненных настроек необходимо перенести файл **setenv.sh** в каталог веб-сервера следующей командой:

sudo mv setenv.sh /opt/tomcat-wfm/bin/setenv.sh

Далее, в файле каталога ROOT/js/config.js необходимо отредактировать строку конфигурации связи с бэкэндом, в данном случае требуется прописать внешний адрес текущего сервера:

backend: 'https://ip\_addres:port/wfmserver/api/'

После сохранения изменений, требуется выполнить перенос всего каталога ROOT в директорию с веб-сервером командой:

sudo mv ROOT /opt/tomcat-wfm/webapps/ROOT, с заменой существующего каталога в директории веб-сервера ROOT.

Далее необходимо перенести оставшиеся архивы с расширением **.war** в директорию с веб сервером при помощи следующих команд:

sudo mv ROOT.war /opt/tomcat-wfm/ROOT.war

sudo mv wfmserver.war /opt/tomcat-wfm/wfmserver.war

Порт 8080 web-сервер по умолчанию, порт 8443 web-сервер с SSL по умолчанию. В качестве IP-адреса указывается внешний адрес сервера приложений. В файле /opt/tomcat-wfm/bin/setenv.sh также должен быть указан верный путь до java.

После выполненных настроек можно запускать приложение используя следующую команду:

sudo sh /opt/tomcat-wfm/bin/startup.sh

Журналирование работы веб-сервера осуществляется в специальном файле, расположенном по адресу /opt/tomcat-wfm/logs/catalina.out

Для того что бы убедиться в корректности внесенных настроек необходимо выполнить следующую команду для просмотра логов работы веб-сервера:

sudo cat /opt/tomcat-wfm/logs/catalina.out

Во время первого запуска приложения wfm-server, в шаблоне, созданном ранее под базу данных WFM указанном в конфигурационном файле **wfmserver.properties**, будут развернуты все необходимы для работы приложения схемы и таблицы, дополнительных действий не требуется.

Последним шагом требуется установить и настроить веб-сервер Nginx, который используется для перенаправления потока запросов на Apache Tomcat.

Для установки веб-сервера Nginx воспользуемся командой:

sudo yum install nginx

После завершения установки необходимо активировать веб-сервер Nginx и добавить его в автозагрузку для того, чтобы веб-сервер Nginx запускался автоматически после перезагрузки сервера. Требуется выполнить следующие команды:

* sudo systemctl start nginx
* sudo systemctl enable nginx

Что бы произвести настройку веб-сервер Nginx следует создать конфигурационный файл следующего содержания:

sudo mcedit /etc/nginx/cond.d/test.conf

server {

gzip on;

gzip\_comp\_level 5;

gzip\_proxied any;

gzip\_static on;

gzip\_types text/plain text/css application/json application/x-javascript text/xml application/xml application/xml+rss text/javascript application/javascript;

client\_max\_body\_size 100M;

listen 443 ssl;

server\_name test-wfmserver.ru;

ssl\_certificate /etc/nginx/cert/testru\_bundle.crt;

ssl\_certificate\_key /etc/nginx/cert/certificate.key;

location / {

# ... various proxy headers, then ...

proxy\_set\_header X-Real-IP $remote\_addr;

proxy\_set\_header X-Forwarded-For $proxy\_add\_x\_forwarded\_for;

proxy\_set\_header Host $http\_host;

proxy\_pass http://127.0.0.1:8082;

proxy\_redirect off;

proxy\_connect\_timeout 900;

proxy\_send\_timeout 900;

proxy\_read\_timeout 900;

send\_timeout 900;

}

}

После сохранения изменений в конфигурационном файле /etc/nginx/conf.d/test.ru настройка производительного веб-сервера Nginx закончена.

* 1. Настройка универсального интеграционного адаптера

Универсальный интеграционный адаптер представляет из себя специальное веб-приложение, упакованное в формат архива с расширением .war в соответствии со спецификацией Java-сервлетов. Выполняется такое приложение на веб-сервере и содержит в своем составе дескриптор развертывания "web.xml", определяющий все свойства веб-приложения. Основными задачами интеграционного адаптера является:

* организация взаимодействия между системами на основе сервисов, включая их публикацию, объединение и использование;
* взаимодействие с различными источниками данных и информационными системами;
* унификация структуры полученных разнородных данных;
* обеспечение гибкости и скорости реализации новых сценариев взаимодействия.

В текущей версии универсального интеграционного адаптера реализовано взаимодействие между WFM системой и Cisco CCX / CUCM – телефония контактного центра, а также реализована возможность импорта данных из базы HPSM – система управления ИТ сервисами.

Система управления телефонией контактного центра - является системой поставщиком данных для WFM в части передачи информации о количестве поступивших звонков, среднем времени их обработки, а также сведения тематической составляющей телефонного обращения.

Система управления ИТ-сервисами, предоставляет WFM сведения о текущих статусах сотрудников.

Настройка универсального интеграционного адаптера, а именно конфигурация подключения к системам поставщикам данных производится в файле tomcat-connector/conf/connector.json, который имеет вид:

{

"consumers": [ *//Наименование массива*

{

"id": 1, *// Идентификатор*

"type": "wfm", *//Тип*

"host": "http://\*\*.\*\*.\*\*\*.\*\*:8080/wfmserver", *//Адрес сервера, где расположен адаптер*

"user": "username", *// Пользователь*

"password": "password" *//Пароль*

}

],

"systems": [],

"databases": [ *//Наименование массива с настройками подключения к мастер-системам*

{

"id": 1, *//Идентификатор*

"type": "\*\*\*-statistic", *//Тип*

"host": "host", *//Адрес мастер системы*

"port": "port", *//Порт мастер системы*

"user": "username", *//Пользователь для подключения к мастер системе*

"password": "password", *//Пароль для подключения к мастер системе*

"databaseName": "SMInfo", *// Наименование базы данных*

"from": "2019-01-01", *// Начала периода сбора данных*

"to": "2019-12-31", *//Окончание периода сбора данных*

"splitEvent":true, *// Признак преобразования данных в 15 минут*

"viewName": "[dbo].[\*Наименование представления\*]" *//Наименование представления*

}.

Если же требуется осуществить подключение новой мастер системы, то необходимо выполнить изменение конфигурации универсального интеграционного адаптера, при необходимости проведения работ Исполнитель необходимо обратиться к Руководителю проекта/Техническую поддержку.

* 1. Перенос данных с тестового контура на продуктивный

В случае выбора распределенной архитектуры системы, когда сервер приложений и система управления базами данных устанавливаются на разных серверных мощностях единственным и ключевым требованием для работоспособности, системы является организация сетевой доступности между сервером приложений и сервером, на котором планируется установка системы управления базами данных.

Для осуществления переноса данных с тестового контура в продуктивный требуется выполнить следующие операции:

Создать архив командной каталога на тестовом контуре

zip -r tomcat-wfm.zip /opt/tomcat-wfm

где

**zip** – команда архиватора

**-r** – Рекурсивно заархивировать все содержимое каталога tomcat-wfm

**tomcat-wfm** – наименование полученного архива

**/opt/tomcat-wfm** – путь до каталога, который необходимо заархивировать

zip -r tomcat-connector.zip opt/tomcat-connector

Далее требуется осуществить перенос созданных архивов на продуктивный сервер, для этого требуется использовать утилиты защищенного копирования файлов между хостами по протоколу SSH - SCP (Secure Copy Protocol). Команда имеет вид:

scp tomcat-wfm.zip user@host:/opt/ и scp tomcat-connector.zip user@host:/opt/

Следующим шагом требуется создать резервную копию базы данных

sudo pg\_dump -Fc --host=localhost --username=tomcat --dbname=gpn\_wfm -f gpn\_wfm.pgsqldump

После создания резервной копии (дампа) базы данных, требуется осуществить ее перенос на сервер продуктивного контура, где планируется развернуть базу данных системы WFM. Для этого также следует использовать утилиту SCP:

scp gpn\_wfm.pgsqldump user@host\_database:/tmp/

Следующим шагом требуется подключиться к серверу базы данных продуктивного контура и осуществить восстановление базы данных из перенесенной ранее резервной копии.

Для этого подключаемся к серверу БД по SSH

ssh wfm@host\_database

Переключение на пользователя root

root -i

Переключение на пользователя postgres

su – postgres

Создание образца базы данных

psql create database gpn\_wfm owner = tomcat;

Восстановление базы данных из резервной копии

pg\_restore -d gpn\_wfm -Fc -v /tmp/gpn\_wfm.pgsqldump

На этом восстановление и настройка сервера базы данных продуктивного контура закончена.

Далее требуется осуществить настройку сервера приложений продуктивного контура, для этого требуется подключиться по ssh

ssh wfm@host\_app

Перейти в директорию с ранее перенесенными архивами

cd /opt

Выполнить разархивирование файлов:

* unzip tomcat-wfm
* unzip tomcat-connector

Отредактировать строку в файле tomcat-wfm/conf/wfmserver.properties

dataSource.url=jdbc:postgresql://localhost:5432/gpn\_wfm

где необходимо вместо "localhost" прописать адрес сервера базы данных продуктивного контура.

Для редактирования файла можно использовать команду

mcedit tomcat-wfm/conf/wfmserver.properties

Следующим шагом следует отредактировать файл конфигурации интеграционного адаптера, расположенный в директории **tomcat-connector/conf/connector.json**. Необходимо изменить массив данных **consumer**, поле **host** прописать адрес текущего сервера приложений продуктивного контура

{

"consumers": [

{

"id": 1,

"type": "wfm",

"host": "http://текущий\_адрес\_сервер/wfmserver",

"user": "username",

"password": "password"

}

Последним шагом необходимо выполнить запуск интеграционного адаптера и сервера приложений, для этого требуется выполнить следующие команды находясь в директории **/opt**

* sh tomcat-wfm/bin/startup.sh
* sh tomcat-connector/bin/startup.sh
  1. Создание произвольных отчетных форм через конструктора отчетов

Использование конструктора отчетов предполагает наличие у пользователя начального уровня знаний языка структурированных запросов (SQL).

Для того, чтобы создать отчет с агрегацией по одному часу необходимо авторизоваться в веб интерфейсе систем и выполнить следующие действия:

* Выбрать пункт главного меню “Отчеты” - “Добавить отчет”
* Задать название, например,- “Отчет по WAPE за 15 минут”
* В поле запроса вставить следующие данные

|  |
| --- |
| select  @dt1 || ' - ' || @dt2 Период,  q.name Очередь,  --fc.start,  --sum(fc.calls) forecast,  --sum(ch.calls) history,  --abs(sum(fc.calls)-sum(ch.calls)),  --sum(fc.calls)-sum(ch.calls),  round(abs(sum(abs(fc.calls -ch.calls)))/sum(ch.calls)::numeric\*100,2) "Показатель, %"  from calls\_history ch  full join forecast fc on ch.start=fc.start and ch.queue\_id=fc.queue\_id  inner join queue q on q.id=fc.queue\_id  where fc.start is not null  and fc.start >=@dt1 ::TIMESTAMP and fc.start<=@dt2 ::TIMESTAMP  group by 1,2  having abs(sum(fc.calls)-sum(ch.calls)) is not null and abs(sum(fc.calls)-sum(ch.calls))<>0  order by 1; |

* Добавить строки и названия полей согласно рисунку ниже

Изображение выглядит как снимок экрана

Автоматически созданное описание

Рисунок 1. Добавления строки и название полей

* Сохранить и выполнить отчет

В результате отобразится отчетная форма по показателю WAPE по 15 минут за выбранный интервал по всем очередям этого интервала, где есть прогнозные и фактические данные.

1. Настройка типов авторизации

В системе существует два возможных типа авторизации, при помощи учётных записей из Active Directory и при помощи учётных записей системы WFM.

Режим работы настраивается через конфигурационный файл, расположенный на сервере приложения по абсолютному пути **/opt/tomcat-wfm/conf/wfmserver.properties**. Подробное описание указанного конфигурационного файла приведено в приложении 1.

В данном файле за тип авторизации отвечает параметр **ldap.enable**, при значении **True** система будет пытаться авторизовать пользователя по протоколу **ldap** на сервере, указанному ниже в конфигурационном файле. При значении **False** логин и пароль будут проверяться по локальной базе данных.

* 1. Журналирование событий

При работе системы WFM предусмотрено журналирование событий на следующих уровнях:

1. уровень ОС:

– события операционной системы Linux, связанные с авторизацией пользователей, функционированием системных служб, неполадках в работе ОС и др.;

2. уровень СУБД:

– протоколы работы СУБД PostgreSQL;

3. прикладной уровень:

– журнал событий, связанных с действиями пользователей и работой программного обеспечения (хранится в базе данных системы WFM);

– протоколы работы программных компонентов системы (log-файлы).

Ниже приведена более подробная информация об указанных журналах событий.

Журналы событий ОС

На каждом сервере инсталляции системы WFM располагаются стандартные журналы работы операционной системы Linux, которые содержат события, связанные с авторизацией пользователей операционной системы, а также ошибками в работе приложений.

Файлы журналов операционной системы располагаются в стандартной директории /var/log, наиболее важными и информативными являются файлы:

• **/var/log/messages** - содержит глобальный системный журнал, в котором пишутся сообщения с момента запуска системы, сообщения от ядра Linux, различных служб, обнаруженных устройств, сетевых интерфейсов и другое. Более подробную информацию можно найти в официальной документации операционной системы.

• **/var/log/secure** [- информация об авторизации пользователей, включая удачные и неудачные попытки входа в операционную систему, а также задействованные механизмы аутентификации. Более подробную информацию можно найти в официальной документации операционной системы.](https://access.redhat.com/documentation/en-us/red_hat_enterprise_linux/5/html/deployment_guide/ch-logfiles)

Для контроля размера дискового пространства, занимаемого лог файлами, рекомендуемся использовать предустановленную в операционную систему утилиту – Logrotate.

**Журналы СУБД**

В системе используется СУБД PostgreSQL, в которой хранится как информация необходимая для работы системы, так и статистические данные.

Стандартной функцией системы управления базами данных является введение журнала критичных ошибок при манипуляции с данными или объектами БД. СУБД PostgreSQL хранит такие журналы за прошедшую неделю в директории:

**/var/lib/pgsql/12/data/pg\_log/\***

В данном журнале можно найти ошибки в работе приложения с базой данных, такие как ошибки нарушения ссылочной целостности, нарушение доменов атрибута или обращения к объектам базы данных, на которые у пользователя нет прав.

Более подробную информацию можно найти в официальной документации по СУБД:

<https://www.postgresql.org/docs/11/runtime-config-logging.html>

Журналы прикладного уровня

В системе есть 2 журнала, которые связаны непосредственно с работой приложения:

1. Журнал аудита, который хранит события, связанные с действиями пользователей и изменением прав и ролей. Журнал реализован в виде таблицы в базе данных. Данный журнал содержит записи о событиях, которые требуется по регламентам информационной безопасности.

2. Журнал работы java-приложения, который хранится по абсолютному пути: **/opt/tomcat-wfm/logs**, данный журнал содержит служебную информацию о работе приложения и может быть использован для отслеживания различных событий, в том числе критичных ошибок в работе веб-сервера системы.

Журнал аудита хранится в базе данных системы, в таблице audit\_log, типы событий, данные о которых попадают в журнал перечислены в таблице 4.

Таблица 3. Типы событий, данные о которых попадают в журнал аудита системы WFM

| Название события | Описание |
| --- | --- |
| FAIL\_LOGIN\_WITH\_MANY\_ID\_FROM\_ONE\_TERMINAL | Неудачные попытки авторизации с одного IP-адреса |
| FAIL\_LOGIN\_WITH\_ONE\_ID\_FROM\_DIFFERENT\_TERMINALS | Неудачные попытки авторизации с различных IP-адресов |
| ATTEMPTING\_TO\_LOG\_IN\_WITH\_A\_LOCKED\_ACCOUNT | Попытка авторизации с заблокированной учетной записью |
| FREQUENT\_LOGIN\_FROM\_MANY\_WORKSTATION | Частые авторизации с различных IP-адресов |
| FREQUENT\_LOGIN\_FROM\_ONE\_WORKSTATION | Частые авторизации с одного IP-адреса |
| ADD\_AND\_DELETE\_RIGHT\_IN\_SHORT\_TIME | Изменение прав пользователя за короткий интервал времени |
| ADD\_AND\_DELETE\_USER\_IN\_SHORT\_TIME | Создание/удаление пользователей за короткий интервал времени |
| WFM\_SERVER\_STARTUP\_FAIL | Неудачный запуск сервера приложений (возникновение каких-либо неполадок, которые могу сказаться на работоспособности системы) |
| WFM\_SERVER\_STARTED | Запуск сервера приложений |
| WFM\_SERVER\_STOPPED | Остановка сервера приложений |
| DELETE\_RIGHT | Удаление прав пользователя |
| UPDATE\_USER | Обновление данных пользователя |
| DELETE\_USER | Удаление пользователя |
| ADD\_RIGHT | Добавление новых прав пользователю |
| ADD\_USER | Добавление нового пользователя в систему |
| LOGOUT | Завершение сессии пользователя |
| LOGIN | Начало новой сессии пользователя(авторизация) |

Все записи по умолчанию сохраняются в таблицу базы данных с названием audit\_log, структура которой описана ниже.

Структура таблицы audit\_log:

* id (UUID) - идентификатор записи
* ip (varchar(255)) - IP-адрес клиента
* comment (varchar(255)) - комментарий, если есть
* created\_at(timestamp) - дата/время события
* request(varchar(1024)) - содержимое тела запроса клиента
* target(UUID) - идентификатор целевого пользователя
* log\_type(varchar(50)) - тип события
* author\_id(UUID) - идентификатор автора события

Пример заполнения таблицы представлен на Рисунке 1

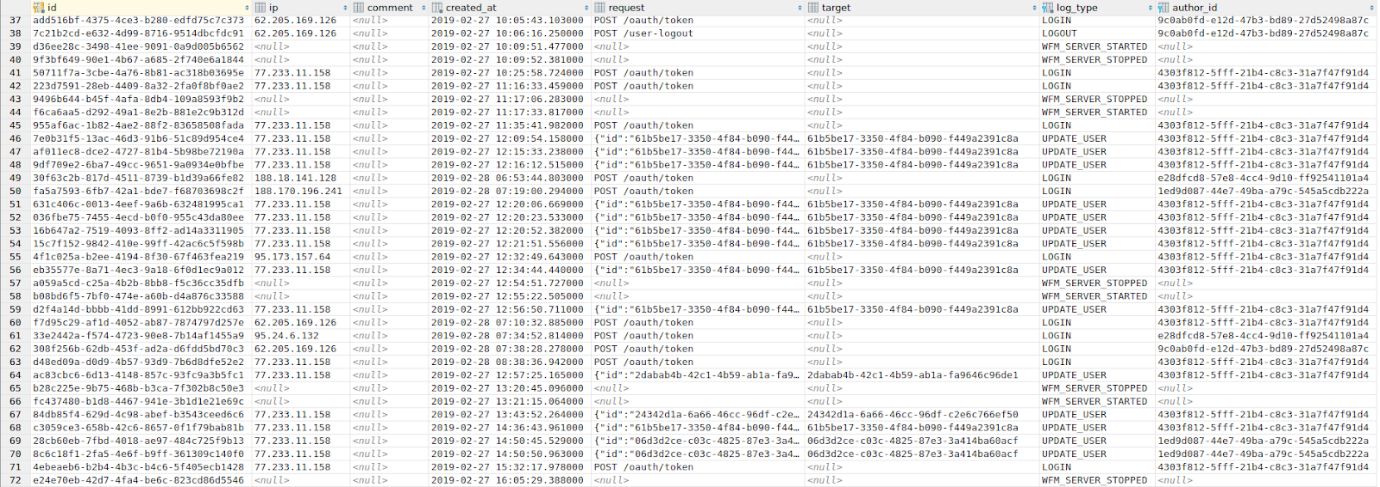


Рисунок 2. Пример заполнения audit\_log

Для детализации информации по попыткам получения несанкционированного доступа в систему существует отдельная таблица, в которой хранятся все попытки авторизации с результатом. Это необходимо для анализа и формирования отчётности по событиям, связанным с попытками входа в систему. По данной таблице можно будет сделать выводы о попытках входа под разными учётными записями с одного IP-адреса, о попытках входа под одной учётной записью с разных IP-адресов и других событиях связанными с несколькими попытками входа. Таблица с детализацией попыток входа называется auth\_audit\_log и находится в той же базе данных. Структура таблицы приведена ниже.

Структура таблицы auth\_audit\_log:

* id(UUID) - Идентификатор записи
* created\_at(timestamp) - Время события
* ip(varchar(255)) - IP-адрес удаленной машины
* logged\_in(boolean) - Успешность авторизации(true/false)
* user\_name(varchar(255)) - имя пользователя

Запись в логе создается при каждой попытке авторизации в системе. В случае, если авторизация успешна в поле logged\_in проставляется значение True, в случае если авторизация не успешна в это же поле проставляется значение False.

В системе существует возможность влиять на часть параметров, записываемых в журнал аудита системы. Параметры журналирования меняются в зависимости от настроек, которые редактируются в файле /opt/tomcat-wfm/conf/wfmserver.properties, который находится на сервере с основным приложением системы. При изменении параметров в вышеуказанном файле, требуется перезапуск всего приложения. Список настроек модуля аудита, которые можно изменять, приведён в таблице ниже.

Таблица 4. Список изменяемых настроек модуля аудита.

| Параметр | Допустимые значения | Описание |
| --- | --- | --- |
| log.lockedUserAuditLogFilter.enable | True/False | Включение/выключение события ATTEMPTING\_TO\_LOG\_IN\_WITH\_A\_LOCKED\_ACCOUNT |
| log.failManyUserOneDeviceAuditLogFilter.enable | True/False | Влючение/отключение события FAIL\_LOGIN\_WITH\_MANY\_ID\_FROM\_ONE\_TERMINAL |
| log.failManyUserOneDeviceAuditLogFilter.checkIntervalMills | integer | Период в мс, за который считается количество попыток по событию FAIL\_LOGIN\_WITH\_MANY\_ID\_FROM\_ONE\_TERMINAL |
| log.failManyUserOneDeviceAuditLogFilter.maxUserNamesCount | integer | Максимальное количество пользователей, пытавшихся авторизоваться с одного устройства за интервал log.failManyUserOneDeviceAuditLogFilter.checkIntervalMills |
| log.failManyUserOneDeviceAuditLogFilter.maxAttemptsCount= | integer | Максимальное количество попыток без логирования по событию FAIL\_LOGIN\_WITH\_MANY\_ID\_FROM\_ONE\_TERMINAL |
| log.failOneUserManyDeviceAuditLogFilter.enable | True/False | Включение/выключение события FAIL\_LOGIN\_WITH\_ONE\_ID\_FROM\_DIFFERENT\_TERMINALS |
| log.failOneUserManyDeviceAuditLogFilter.checkIntervalMills | integer | Период в мс, за который считается количество попыток по событию FAIL\_LOGIN\_WITH\_ONE\_ID\_FROM\_DIFFERENT\_TERMINALS |
| log.failOneUserManyDeviceAuditLogFilter.maxDevicesCount | integer | Максимальное количество различных устройств за интервал, для которых не срабатывает событие FAIL\_LOGIN\_WITH\_ONE\_ID\_FROM\_DIFFERENT\_TERMINALS |
| log.failOneUserManyDeviceAuditLogFilter.maxAttemptsCount | Integer | Максимальное количество попыток без логирования за интервал, по событию FAIL\_LOGIN\_WITH\_ONE\_ID\_FROM\_DIFFERENT\_TERMINALS |
| log.manyUsersOneDeviceAuditLogFilter.enable | True/False | Включение/выключение события FREQUENT\_LOGIN\_FROM\_ONE\_WORKSTATION |
| log.manyUsersOneDeviceAuditLogFilter.checkIntervalMills | integer | Период в мс, за который считается количество попыток по событию FREQUENT\_LOGIN\_FROM\_ONE\_WORKSTATION |
| log.manyUsersOneDeviceAuditLogFilter.maxUserNamesCount | ineteger | Максимальное количество пользователей без логгирования за интервал по событию FREQUENT\_LOGIN\_FROM\_ONE\_WORKSTATION |
| log.oneUserManyDeviceAuditLogFilter.enable | True/False | Включение/выключение события FREQUENT\_LOGIN\_FROM\_MANY\_WORKSTATION |
| log.oneUserManyDeviceAuditLogFilter.checkIntervalMills | Integer | Период в мс, за который считается количество попыток по событию FREQUENT\_LOGIN\_FROM\_MANY\_WORKSTATION |
| log.oneUserManyDeviceAuditLogFilter.maxDevicesCount | integer | Максимальное количество различных устройств без логирования за интервал по событию FREQUENT\_LOGIN\_FROM\_MANY\_WORKSTATION |
| log.authAuditLogListener.login.enable | True/False | Включение/выключение события LOGIN |
| log.authAuditLogListener.logout.enable | True/False | Включение/выключение события LOGOUT |
| log.addAndRemoveUserInShortTimeAuditLogFilter.enable | True/False | Включение/выключение события ADD\_AND\_DELETE\_USER\_IN\_SHORT\_TIME |
| log.addAndRemoveUserInShortTimeAuditLogFilter.checkIntervalMills | integer | Период в мс, за который считаются события ADD\_AND\_DELETE\_USER\_IN\_SHORT\_TIME |
| log.changeUserLogListener.addUserType.enable | True/False | Включение/выключение события ADD\_USER |
| log.changeUserLogListener.deleteUserType.enable | True/False | Включение/выключение события DELETE\_USER |
| log.changeUserLogListener.updateUserType.enable | True/False | Включение/выключение события UPDATE\_USER |
| log.addAndRemoveRightsInShortTimeAuditLogFilter.enable | True/False | Включение/выключение события ADD\_AND\_DELETE\_RIGHT\_IN\_SHORT\_TIME |
| log.addAndRemoveRightsInShortTimeAuditLogFilter.checkIntervalMills | Integer | Период в мс, за который считаются случаи по событию ADD\_AND\_DELETE\_RIGHT\_IN\_SHORT\_TIME |
| log.addAndRemoveRightsInShortTimeAuditLogFilter.deleteRightLogType.enable | True/False | Включение/выключение события удаления прав DELETE\_RIGHT |
| log.addAndRemoveRightsInShortTimeAuditLogFilter.addRightLogType.enable | True/False | Включение/выключение события добавления прав ADD\_RIGHT |
| log.auditLogServiceImpl.maxLengthRequestBody | integer | максимальная длина поля для хранения тела HTTP-запроса в БД |
| log.appAuditLogListener.startedType.enable | True/False | Включение/выключение события WFM\_SERVER\_STARTED |
| log.appAuditLogListener.stoppedType.enable | True/False | Включение/выключение события WFM\_SERVER\_STOPPED |
| log.appAuditLogListener.startupFailType.enable | True/False | Включение/выключение события события WFM\_SERVER\_STARTUP\_FAIL |

Также реализован дополнительный функционал формирования специальных лог файлов по согласованному формату для дальнейшего его анализа специализированными системами.

Набор данных и регистрируемые события:

* дата и время входа (выхода) субъекта доступа в систему (из системы) или загрузки (останова) системы;
* результат попытки входа (аутентификации): успешная или неуспешная;
* идентификатор (логин) субъекта, предъявленный при неуспешной попытке;
* идентификатор (логин) субъекта, предъявленный при попытке доступа;
* дата и время изменения роли, идентификатор (логин) субъект;
* дата и время изменения полномочий субъекта, идентификатор (логин) субъекта изменяемого полномочия, идентификатор (логин) субъекта изменяющего полномочия.

Для создания такого лог файла используется сценарий командной строки для оболочки bash, запуск которого осуществляется с помощью программы периодического выполнения заданий Cron .

На рисунке 6 представлено содержание такого файла сценария.

Изображение выглядит как текст

Автоматически созданное описание

Рисунок 3. Содержание файла сценария

где

#!/bin/bash – путь к bash-интерпретатору

OUTPUT\_PATH – путь, к созданному лог файлу

OUTPUT\_FILENAME\_PREFIX – префикс, используемый в названии файла

INTERVAL – периодичность выполнения

PGSQL\_PATH – путь к системе управления базами данных

"if [ -z "$1" ]

then

echo "No database name"

exit 1;

fi" – Условие при выполнении файла сценария

DBNAME – наименование базы данных

DATE – формат даты

$PGSQL\_PATH $DBNAME -c "COPY (select created\_at,ip,author,log\_type,target as target\_user,SUBSTRING(request,1,128) as request\_payload,SUBSTRING(comment,1,128) as comment from audit\_security\_report where created\_at>=now()- INTERVAL '$INTERVAL' ORDER BY created\_at) TO '$OUTPUT\_PATH/$OUTPUT\_FILENAME\_PREFIX$DATE.csv' DELIMITER ';' CSV HEADER ;" – выполняемый запрос к базе данных.

* 1. Настройка отправки уведомлений

В системе WFM присутствует функционал уведомления пользователей об изменениях, внесенных в Системе посредством PUSH-уведомлений и отправки соответствующих сообщений на электронную почту сотрудников.

Для того что бы включить уведомления требуется внести соответствующие настройки в основной конфигурационный файл /opt/tomcat-wfm/conf/wfmserver.properties, а именно:

fcm.project\_id - Идентификатор проекта на Firebase Cloud Messaging

fcm.api\_key - Токен подключения к API отправки push-уведомлений

fcm.host - Сервер с API для отправки push-уведомлений

fcm.port - Порт подключения к API, для отправки push-уведомлений

fcm.enable - Включение/Выключение отправки push-уведомлений

После внесения описанных изменений в основной конфигурационный файл, требуется перезапустить приложение.

* 1. Порядок остановки и запуска Системы

В случае необходимости отключения серверного оборудования, перед этим требуется осуществить корректную остановку сервера приложений, для этого требуется осуществить авторизацию в операционной системе по протоколу ssh и выполнить следующую команду:

sh /opt/tomcat-wfm/bin/shutdown.sh

Для запуска системы требуется выполнить команду запуска:

sh /opt/tomcat-wfm/bin/startup.sh.

Далее необходимо остановить работу системы управления базы данных, для чего требуется выполнить команду:

sudo service postgresql-12 stop

1. Настройка резервного копирования

Резервное копирование производится средствами СУБД PostgreSQL. Резервное копирование производится автоматически по настроенному расписанию в промежуток времени с 00:00 по 06:00 каждый день. Рекомендуемое место хранения резервных копий – внешний дисковый массив. Основное задание резервного копирования обеспечивает 7 точек восстановления: воскресенье ночью будет выполняться полное резервирование, затем каждую ночь инкрементальное до следующего воскресенья. Резервные копии хранятся неделю на внешнем дисковом массиве.

Стратегия резервного копирования гарантирует восстановление данных на определенный момент времени.

Резервное копирование осуществляется средствами клиентского приложения системы управления базы данных Postgresql – pg\_dump, запуск которой осуществляется с помощью программы периодического выполнения заданий Cron, предустановленной в подавляющее большинство UNIX подобных операционных систем. Результатом выполнения резервного копирования является создание следующих файлов:

* 200303\_000020\_wfm.pgsqldump – резервная копия базы данных
* 200303\_000020\_wfm.log – лог файл, содержащий информацию о ходе создания резервной копии.

Наименование и расширение приведенных примеров файлов указывается при вызове программы pg\_dump.

При необходимости произвести восстановление базы данных из резервной копии, необходимо использовать утилиту pg\_restore, пример использования которой описан ниже:

pg\_restore -d gpn\_wfm -Fc -v /путь к файлу резервной копи

После непродолжительного ожидания база данных будет развернута из файла резервной копии.

В случае необходимости создания резервной копии сервера приложений, потребуется выполнить архивирование всей директории /opt/tomcat-wfm. Выполнить это возможно с помощью консольной следующей команды:

zip -r tomcat-wfm\_200901.zip /opt/tomcat-wfm,

где

zip – команда архиватора

-r – Рекурсивно заархивировать все содержимое каталога tomcat-wfm

tomcat-wfm\_200901.zip – наименование полученного архива

/opt/tomcat-wfm – путь до директории, которую необходимо заархивировать

После выполнения указанной команды, в текущем каталоге отобразится полученный архив с рабочей копией сервера приложений WFM системы, для запуска которого потребуется только разархивировать полученный файл.

1. Поиск и устранение неисправностей

Основным источником информации о возможных неисправностях в работе WFM системы служит лог файл контейнера сервлетов Apache-Tomcat, расположенный по адресу /opt/tomcat-wfm/logs/catalina.out. В частности, в подкаталоге tomcat/bin создаются файлы setenv.bat и setenv.sh, а в подкаталоге tomcat/conf файл logback.xml. Файлы setenv.\* в переменной CATALINA\_OPTS в числе прочего устанавливаются параметры загрузки конфигурационного файла logback.xml.

Файл logback.xml определяет конфигурацию логирования. Рассмотрим структуру этого файла. Элементы appender задают "устройства вывода" логов. Основными аппендерами являются FILE и CONSOLE. В параметре level элемента filter можно задать порог уровня сообщения. По умолчанию порог для файла - DEBUG, для консоли - INFO.

Это означает, что в файл выводятся сообщения с уровнями:

* ERROR;
* WARN;
* INFO;
* DEBUG,

а в консоль - с уровнями:

* ERROR;
* WARN;
* INFO.

Для файлового аппендера в параметре file задается путь к файлу лога.

По умолчанию это файл tomcat/logs/app.log. Элементы logger задают параметры логгеров, через которые производится посылка сообщений из кода программы. Имена логгеров иерархические, то есть например настройки для логгера com.company.sample влияют на логгеры com.company.sample.core.CustomerServiceBean, com.company.sample.web.CustomerBrowse, если для них явно не заданы собственные настройки. Минимальный уровень указывается в атрибуте level. Например, если для логгера задан приоритет INFO, то сообщения с уровнями DEBUG и TRACE выводиться не будут. Следует иметь в виду, что на вывод сообщения также влияет порог уровня, заданный в аппендере. Оперативно изменять уровни для логгеров и пороги аппендеров для работающего сервера можно с помощью экрана Administration > Server Log, доступного в веб-клиенте.

Сделанные настройки логирования действуют только в текущем сеансе работы сервера и в файл не сохраняются. Этот экран позволяет также просматривать и загружать файлы логов из каталога журналов сервера tomcat/logs.

Ниже приведены список наиболее часто встречающихся ошибок (исключений) в процессе эксплуатации WFM системы.

Таблица 5. Список ошибок

| **Название ошибки** | **Описание** |
| --- | --- |
| SocketTimeoutException | Данное исключение появляется, когда отсутствует подключение к сопрягаемой системе. В таком случае требуется проверить настройки коннектора и убедиться в доступности системы поставщика данных с сервера приложений WFM системы |
| PSQLException | Это исключение появляется при отсутствии связи между сервером приложений и базой данных. При возникновении такого исключения требуется проверить состояние службы postgresql. Что бы отобразить текущее состояние службы postgresql требуется выполнить команду в консоли systemctl status postgresql-\*. |
| BeanCreationException | Появление этого исключения, информирует о несовместимости текущей версии коннектора с версией сопрягаемой системы. |

1. проверка функционирования системы

Основными компонентами программного обеспечения WFM является контейнер сервлетов с открытым исходным кодом Apache-Tomcat и система управления базами данных PostgreSQL. Если по каким-то причинам, указанные выше службы перестанут быть активными, WFM система перестанет отвечать на запросы пользователей, а именно первым наиболее заметным признаком наличия сбоя в работе WFM системы является недоступность веб-страницы. Подобная ситуация возможна при отключении службы apache-tomcat либо некорректной настройки веб-сервер Nginx, а также остановки работы среды выполнения Java.

Текущее состояние служб, необходимых для корректной работы Системы можно проверить с помощью команды в консоли сервера приложений.

ps -aux | grep java

На рисунке 1 изображен пример корректной работы сервера приложений. Если же Aache-Tomcat не активен, запуск его можно выполнить с помощью команды в консоли "sh /opt/tomcat-wfm/bin/startup.sh".

Изображение выглядит как текст

Автоматически созданное описание

Рисунок 4. Пример работы сервера.

Текущее состояние службы СУБД PostgreSQL можно проверить с помощью консольной команды "systemctl status postgresql-\*". На рисунке 8 изображен пример запущенной службы PostgreSQL.

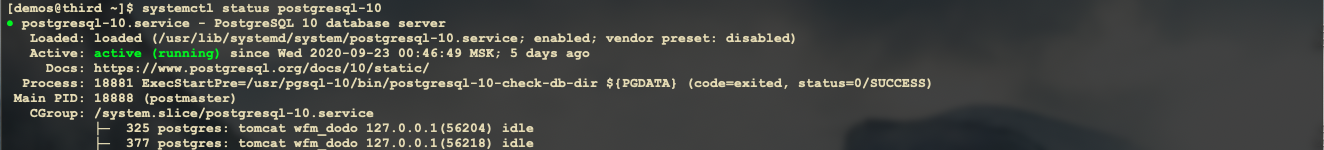


Рисунок 5. Пример запущенной службы PostgreSQL.

При исправной работе указанных служб программное обеспечение WFM гарантировано сохранит свое корректное функционирование.

Более расширенное состояние работы операционной системы сервера приложений и запущенных служб возможно просмотреть используя консольную утилиту htop. На рисунке 3 изображен пример запуска утилиты htop.

В случае, если служба postgresql не активна, следует осуществить ее запуск выполнив в консоли команду "systemctl start postgresql-\*".

Итого для корректной работы системы должны быть запущены следующие сервисы:

Nginx – веб сервер;

Apache-Tomcat – контейнер сервлетов;

PostgreSQL – система управления базами данных.

В процессе установки системы, рекомендуется добавить описанные сервисы в автозагрузку операционной системы, управление которой осуществляется системой инициализации system.

Изображение выглядит как зеленый, металлический, деревянный, стол

Автоматически созданное описание

Рисунок 6. Пример работы htop

Если по каким-либо причинам, не удается обеспечить отказоустойчивое функционирование описанных выше служб, необходимо обратиться в службу технической поддержки.

Приложение 1 ОПИСАНИЕ КОНФИГУРАЦИОННОГО ФАЙЛА СИСТЕМЫ

Файл хранится в формате plain text, где одна строка – это один параметр. Имя параметра располагается слева от знака равно, значение – справа от знака равно. Строки, которые начинаются с символа ‘#’ не учитываются. Для применения изменений требуется перезапуск приложения. Файл располагается на сервере с основным приложением, по абсолютному пути:

/opt/tomcat-wfm/conf/wfmserver.properties.

Таблица 6. Описание конфигурационного файла.

| Имя параметра | Описание параметра | Пример значения |
| --- | --- | --- |
| dataSource.driverClassName | Тип используемой СУБД | org.postgresql.Driver |
| dataSource.url | Строка подключения к СУБД | jdbc:postgresql://localhost:5432/wfm |
| dataSource.username | Пользователь для работы с СУБД | USERNAME |
| dataSource.password | Пароль для работы с СУБД | USER\_PASSWORD |
| hibernate.dialect | Диалект языка работы с БД | com.optisystems.wfm.CustomPgDialect |
| hibernate.show\_sql | Включение/Выключение логирования запросов к СУБД в журнал приложения | False |
| hibernate.hbm2ddl | Управление DDL операциями, при обнаружении изменении в схеме | Update |
| hibernate.jdbc.batch\_size | Количество запросов, которые можно объединить в одно обращение к СУБД | 30 |
| hibernate.generate\_statistics | Включение/Выключение сбора статистики – для возможной оптимизации работы Hibernate | False |
| hibernate.cache.use.query\_cache | Включение/Выключение использования кэша, при работе с СУБД | False |
| hibernate.cache.use\_second\_level\_cache | Включение/Выключение использования кэша второго уровня, при работе с СУБД | False |
| hibernate.max\_pool\_size | Размер пула соединений при работе с СУБД | 10 |
| fcm.project\_id | Идентификатор проекта на Firebase Cloud Messaging | 995859600351 |
| fcm.api\_key | Токен подключения к API отправки push-уведомлений | AAAA593bk98:APA91bFqTUoHG9y1q7thW\_ |
| fcm.host | Сервер с API для отправки push-уведомлений | fcm-xmpp.googleapis.com |
| fcm.port | Порт подключения к API, для отправки push-уведомлений | 5235 |
| fcm.enable | Включение/Выключение отправки push-уведомлений | true |
| oauth.accessTokenLifeTime | Время жизни токена в секундах, получаемого пользователем при успешной авторизации | 3600 |
| oauth.refreshTokenLifeTime | Время жизни продлённого токена, происходит без повторной авторизации | 2592000 |
| oauth.numberOfAuthorizationAttempts | Количество неуспешных попыток авторизации пользователя. После исчерпания которых, пользователь блокируется. | 0 (число попыток неограниченно) |
| spring.jackson.deserialization.ADJUST\_DATES\_TO\_CONTEXT\_TIME\_ZONE | Включение/Выключение конвертации часовых поясов в Spring | False (Запрещено менять) |
| ldap.enable | Включение/Выключение авторизации при помощи Active Directory | false |
| ldap.url | Строка подключения к контроллеру домена, для авторизации по протоколу ldap | ldap://localhost:10389/ |
| ldap.user-dn-patterns | Шаблон создания, DN строки, которая передаётся вовремя авторизации на сервер контроллера домена | uid={0},ou=system |
| ldap.password-attribute | Атрибут, в котором передавать пароль, при авторизации через Active Directory | user\_password |
| ldap.search-base | Место в иерархии AD, от которого начинается поиск сотрудника. | ou=system |
| schedules.scheduleTable.cellViewType | Тип отображения расписания в таблице расписания, по сменам или по активностям. | byShift |

* **Используемые термины и сокращения**

В Таблице приведен перечень терминов и сокращений, используемых в настоящем документе

*Таблица 7 Перечень терминов и сокращений*

|  |  |
| --- | --- |
| **Термин** | **Определение** |
| ИС | Информационная система |
| JSON | Текстовый формат обмена данными, основанный на JavaScript. |
| Система, WFM система | Продукт WFM |
| БД | База данных |
| Аватарка | Графическое представление пользователя |
| БД | База данных |